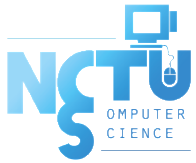
A Preliminary Introduction to Software Security

Chun-Ying Huang (黃俊穎)
chuang@cs.nctu.edu.tw
Dept. of Computer Science
National Yang Ming Chiao Tung University



Outline

- Some Security Facts
- Software Security
- Learning Security
- Summary

Some Security Facts

#1 Security Matters

Security Matters

- In the past
  - Virus
  - Trojan
  - Spam
- Now, we have more …
  - Bots
  - Mobile malware
  - Social engineering – phishing, spear-phishing, APT

BIG Events in Taiwan

2016

2017

2018

BIG Events in Taiwan (Cont'd)

2019

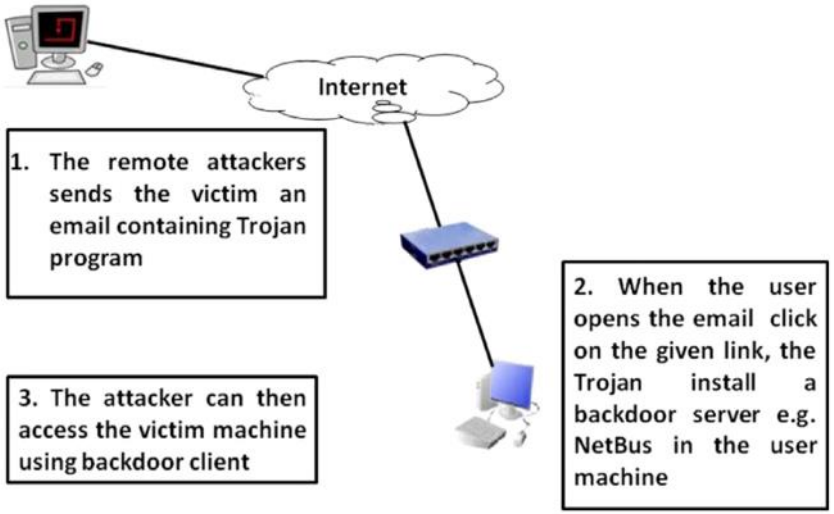
2020

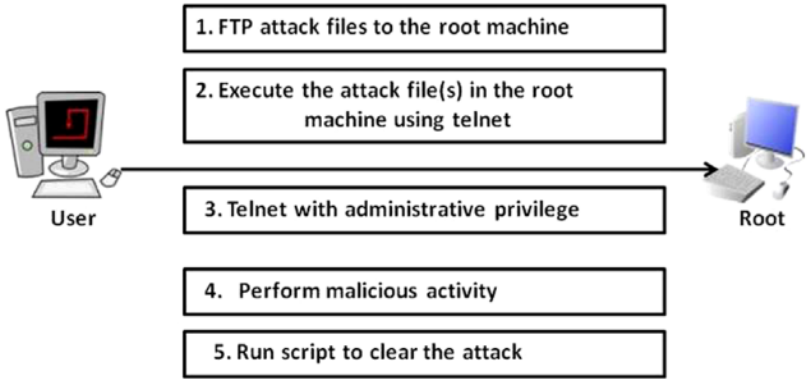
2021

#2 Shift of “Entry Points”

Network Attacks: R2L and U2R

- R2L – from remote to local: Active Attacks!
- U2R – from user to root: Privilege Escalation!





Attacks Shift from Servers to Clients – 2007

- View Points from Bruce Schneier
  - Operating systems have fewer vulnerabilities
  - Significant growth in the number of client-side vulnerabilities
  - Users are allowed to browse Internet in their organizations
  - Web application vulnerabilities account for half of vulnerabilities
  - Attackers are finding more creative ways to obtain sensitive data from organizations

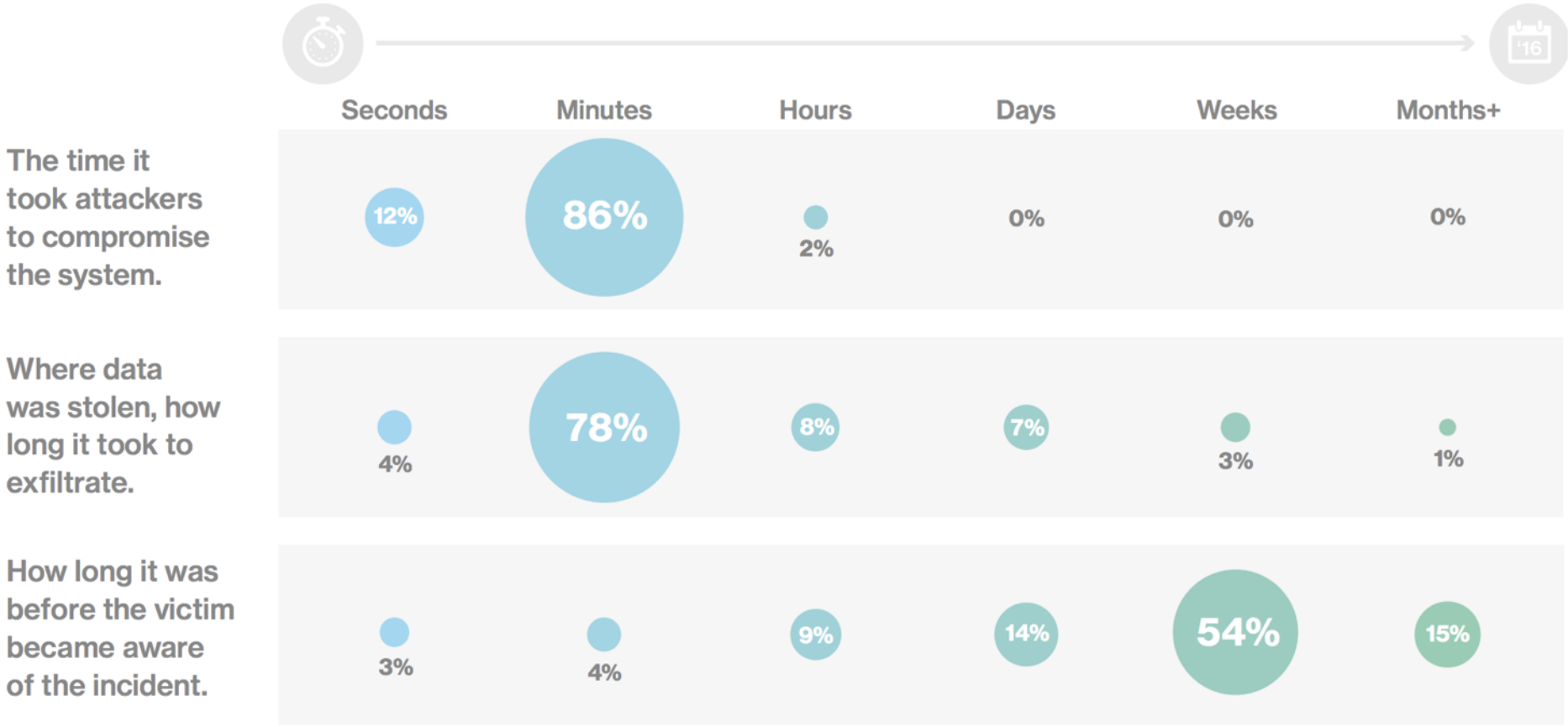
Network Attacks: R2L and U2R

- R2L – from remote to local: Active Attacks!
- U2R – from user to root: Privilege Escalation!

- R2L – from remote to local: Passive Attacks!
- U2R – from user to root: Privilege Escalation!

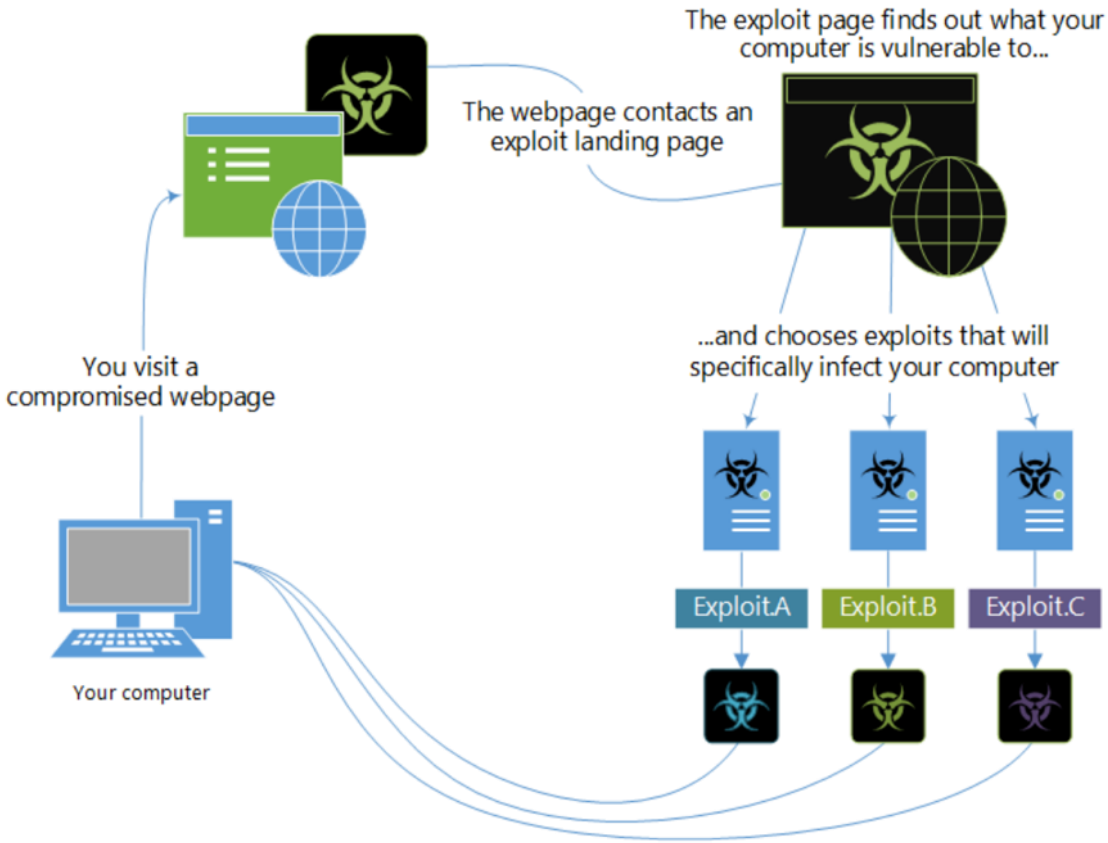
#3 Easy to be Compromised, but Hard to be Detected/Recovered

Incident Timeline for Financial Services



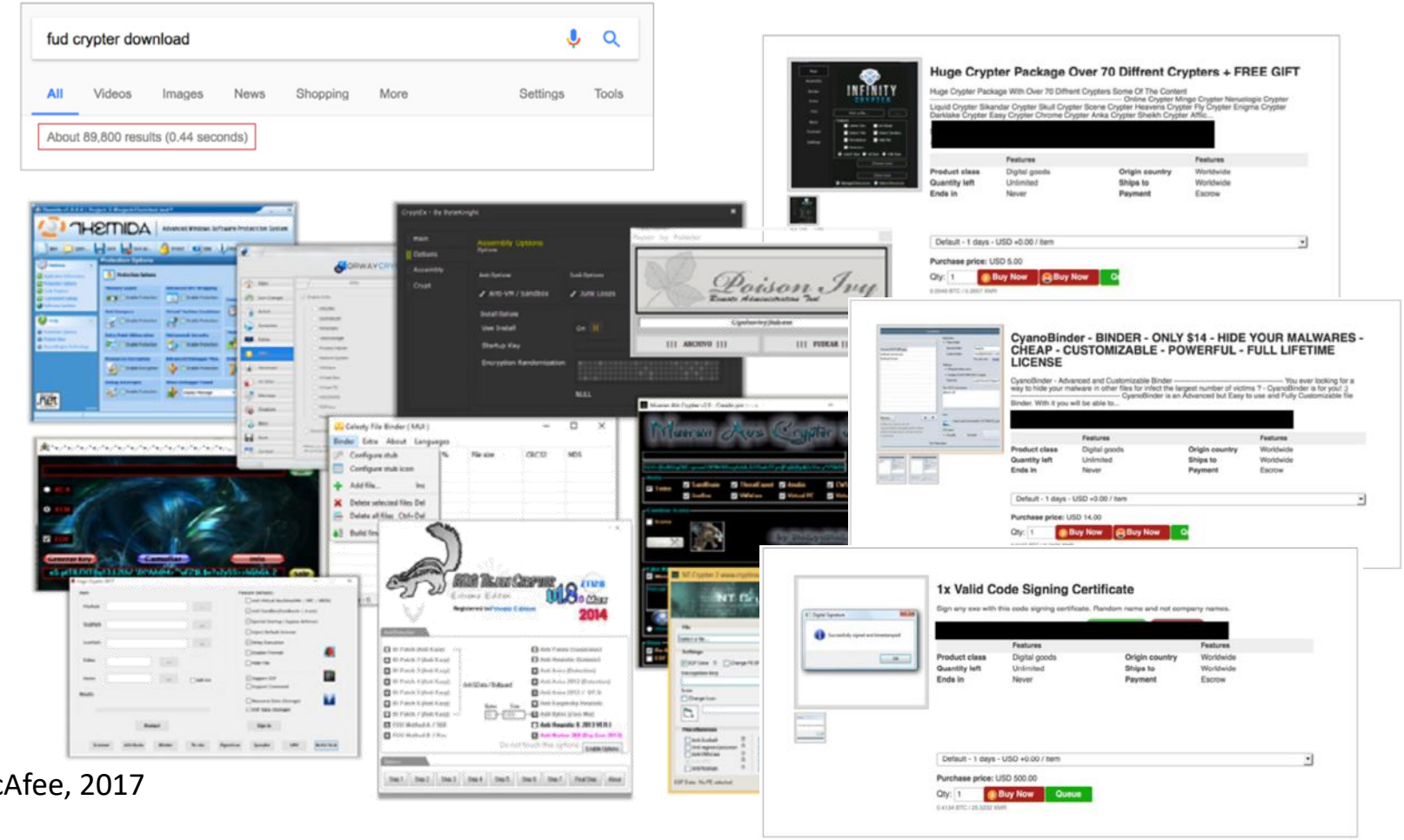
Source: VerizonEnterprise Data Breach Investigations Report, 2016

Exploit Kits are Popular for Hackers



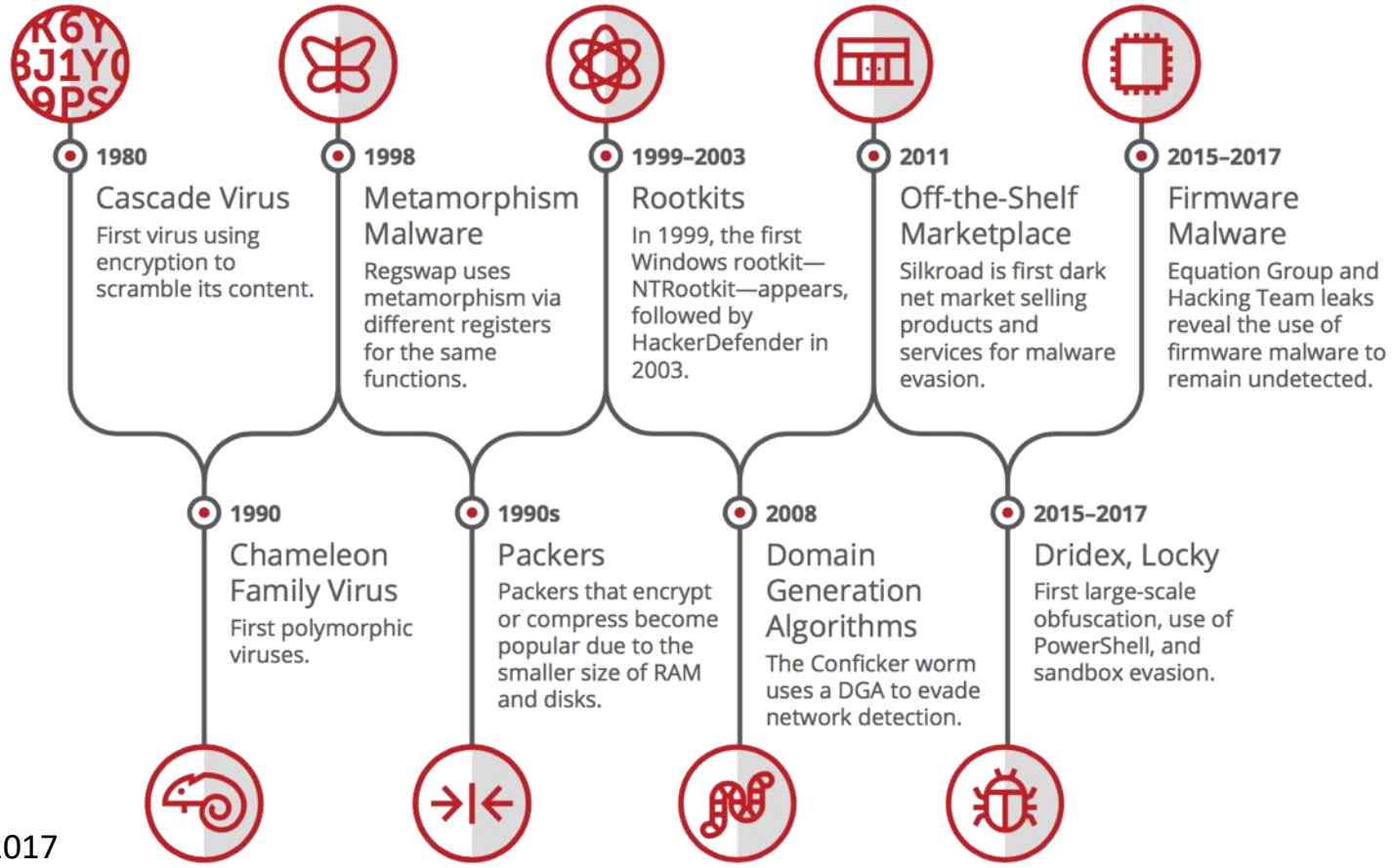
Source: Microsoft Security Intelligent Report Vol. 22, Jan-Mar, 2017

Pay and then Go!



Source: McAfee, 2017

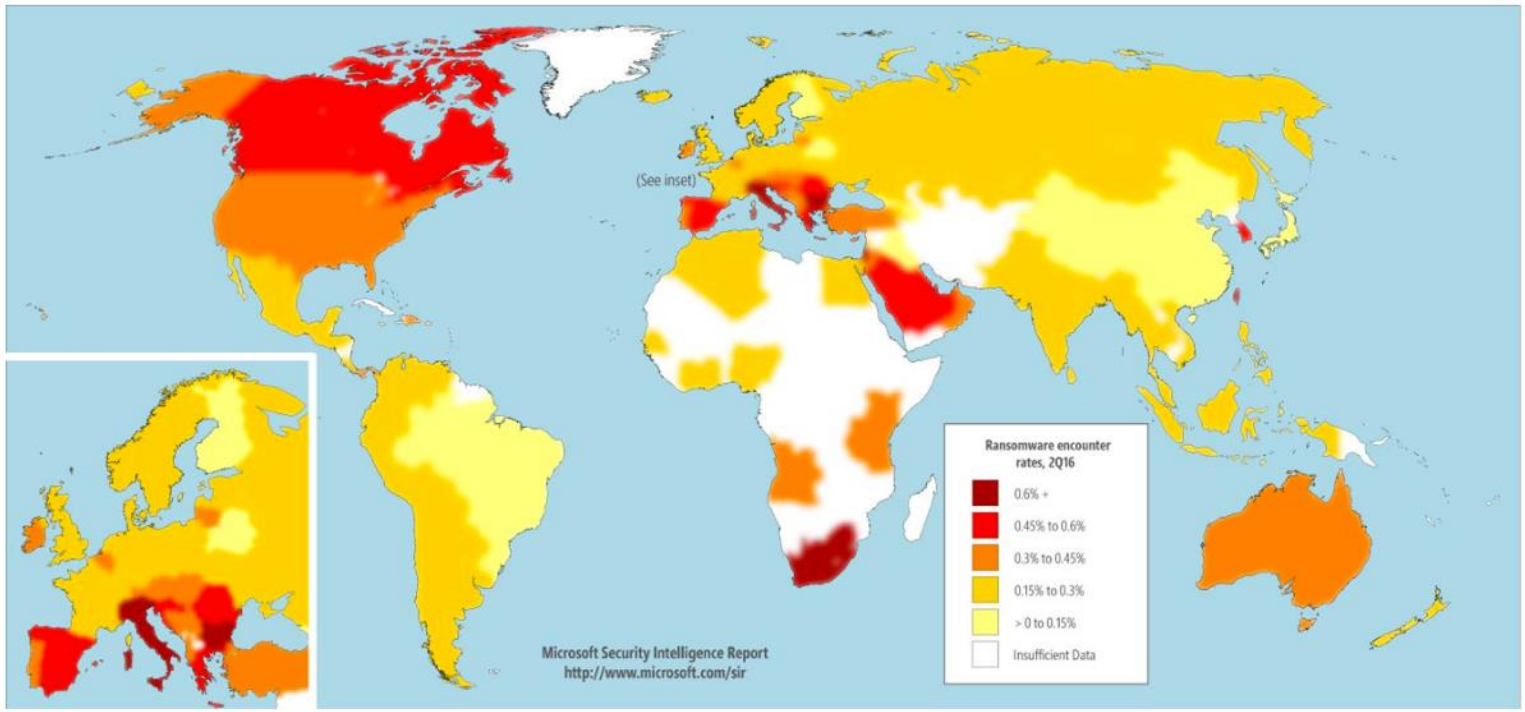
Major Milestones in the Evolution of Evasion Techniques



Source: McAfee, 2017

Hosts in Taiwan are More Vulnerable!? – Ransomware

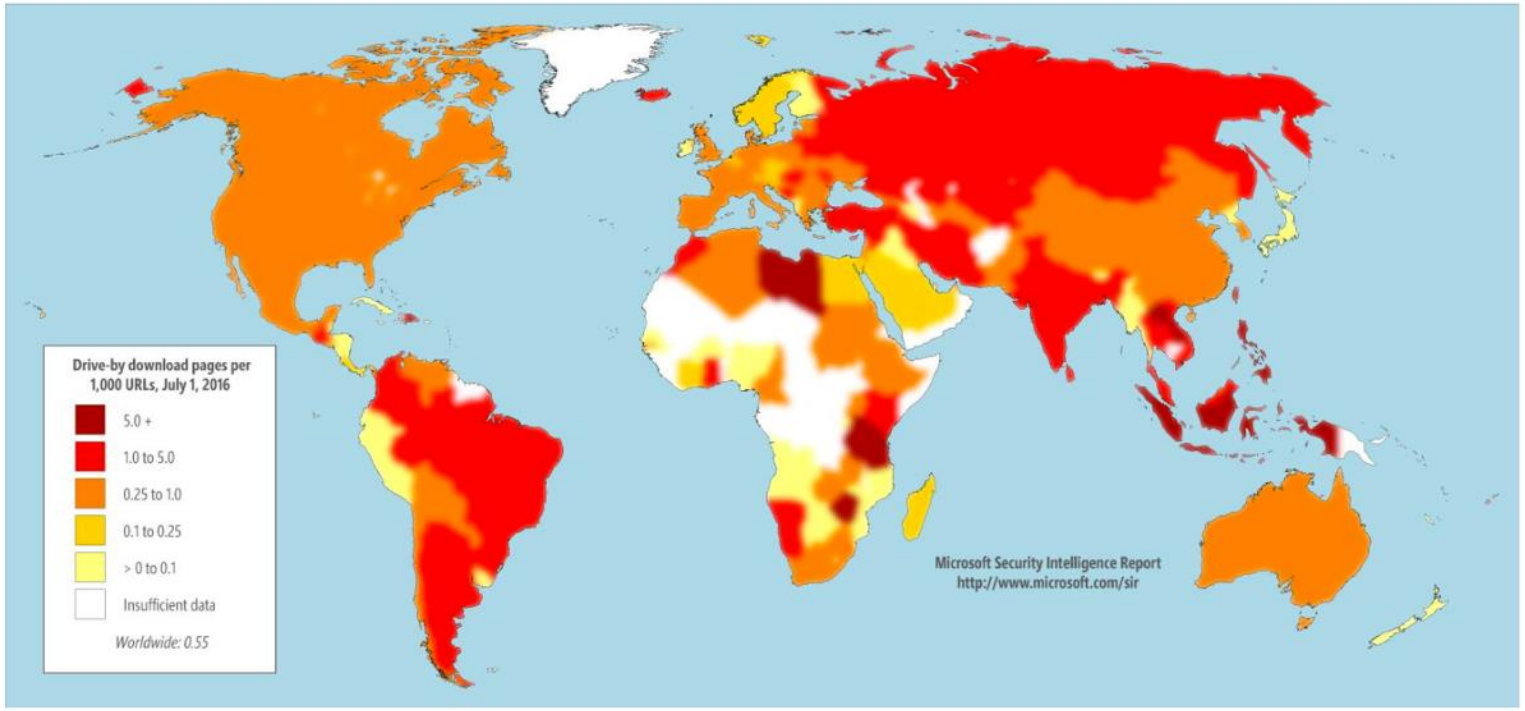
- Top #3 – 0.67% Encounter Rate (2016)



Source: Microsoft Security Intelligent Report Vol. 21, Jan-Jun, 2016

Hosts in Taiwan are More Vulnerable!? – Drive-by-Download Hosts

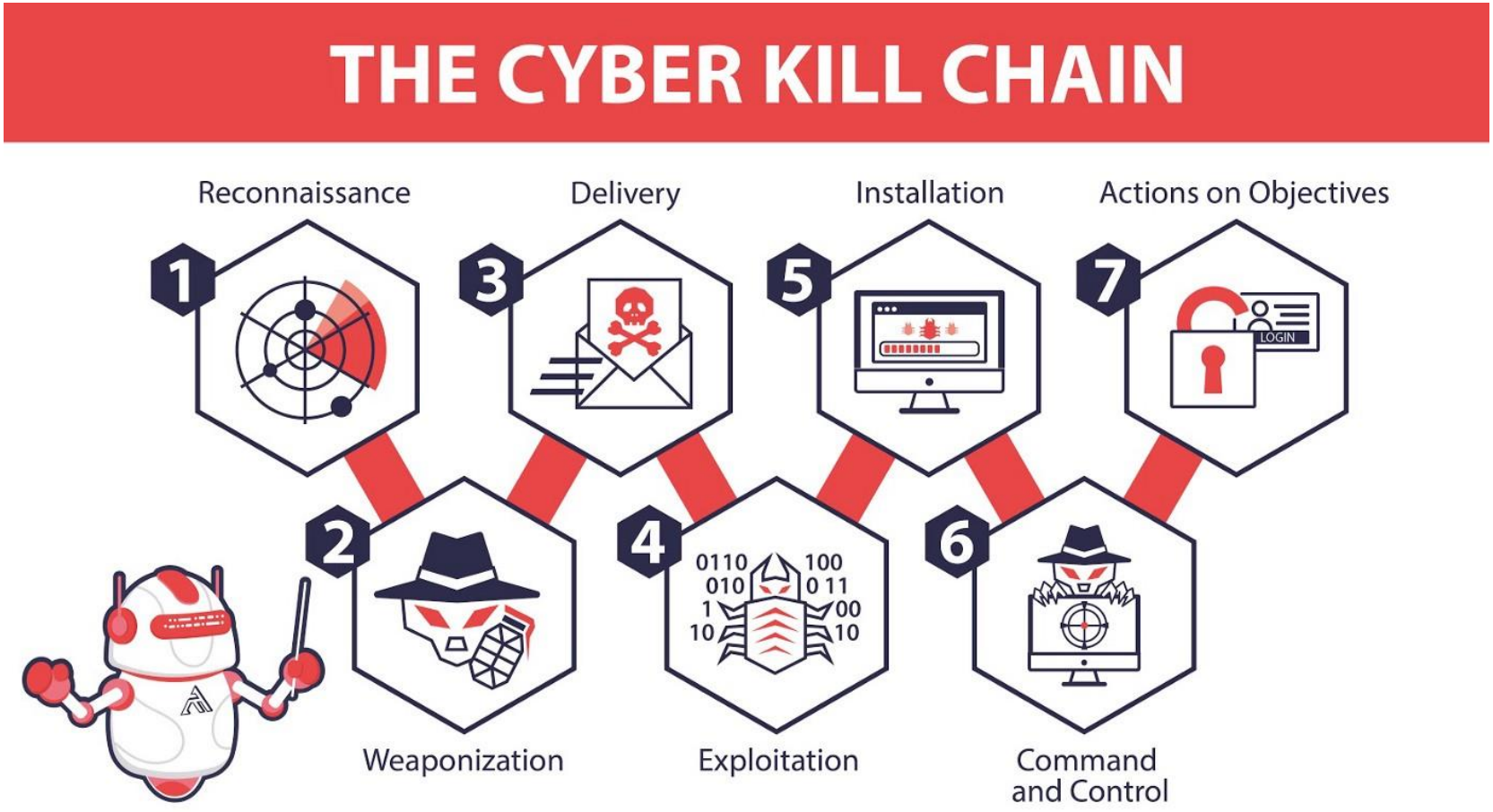
- Top #1 – 7.4 URLs per 1000 URLs (2016 and 2017)



Source: Microsoft Security Intelligent Report Vol. 21/22, Jan-Jun, 2016 and Jan-Mar, 2017

#4 Can be Systematically Handled

How to Attack?



Source: CyCraft @ Medium, 2020
https://medium.com/cycraft/cycraft-classroom-mitre-att-ck-vs-cyber-kill-chain-vs-diamond-model-1cc8fa49a20f

MITRE Att&ck

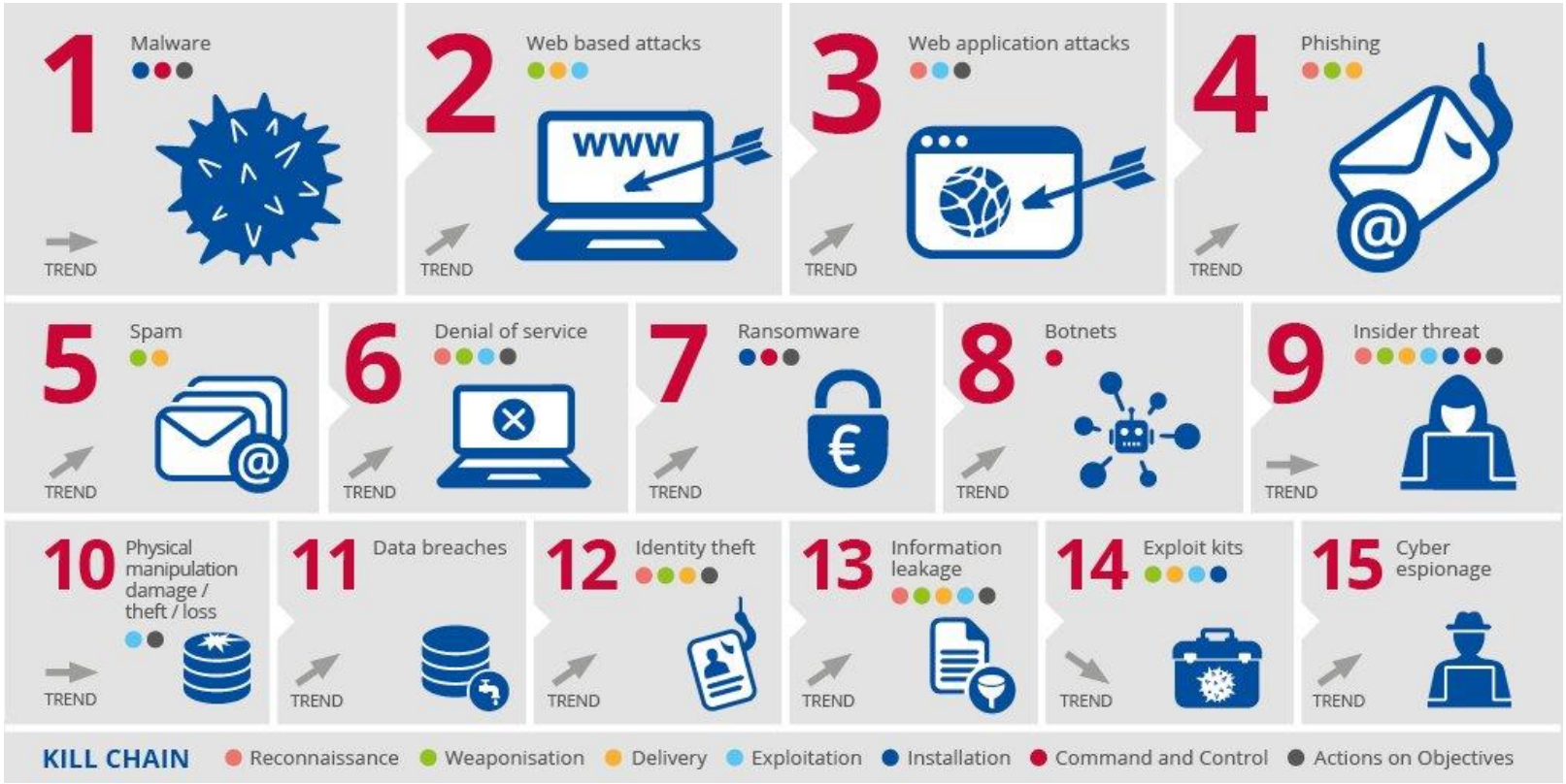

Reconnaissance
10 techniques
Active Scanning (2)
Gather Victim Host Information (4)
Gather Victim Identity Information (3)
Gather Victim Network Information (6)
Gather Victim Org Information (4)
Phishing for Information (3)
Search Closed Sources (2)
Search Open Technical Databases (5)
Search Open Websites/Domains (2)
Search Victim-Owned Websites
Resource Development
7 techniques
Acquire Infrastructure (6)
Compromise Accounts (2)
Compromise Infrastructure (6)
Develop Capabilities (4)
Establish Accounts (2)
Obtain Capabilities (6)
Stage Capabilities (5)
Initial Access
9 techniques
Drive-by Compromise
Exploit Public-Facing Application
External Remote Services
Hardware Additions
Phishing (3)
Replication Through Removable Media
Supply Chain Compromise (3)
Trusted Relationship
Valid Accounts (4)
Execution
12 techniques
Command and Scripting Interpreter (8)
Container Administration Command
Deploy Container
Exploitation for Client Execution
Inter-Process Communication (2)
Native API
Scheduled Task/Job (6)
Shared Modules
Software Deployment Tools
System Services (2)
User Execution (3)
Windows Management Instrumentation
Persistence
19 techniques
Account Manipulation (4)
BITS Jobs
Boot or Logon Autostart Execution (15)
Boot or Logon Initialization Scripts (5)
Browser Extensions
Compromise Client Software Binary
Create Account (3)
Create or Modify System Process (4)
Event Triggered Execution (15)
External Remote Services
Hijack Execution Flow (11)
Implant Internal Image
Modify Authentication Process (4)
Office Application Startup (6)
Pre-OS Boot (5)
Scheduled Task/Job (6)
Server Software Component (4)
Traffic Signaling (1)
Valid Accounts (4)
Privilege Escalation
13 techniques
Abuse Elevation Control Mechanism (4)
Access Token Manipulation (5)
Boot or Logon Autostart Execution (15)
Boot or Logon Initialization Scripts (5)
Create or Modify System Process (4)
Domain Policy Modification (2)
Escape to Host
Event Triggered Execution (15)
Exploitation for Privilege Escalation
Hijack Execution Flow (11)
Process Injection (11)
Scheduled Task/Job (6)
Valid Accounts (4)
Defense Evasion
40 techniques
Abuse Elevation Control Mechanism (4)
Access Token Manipulation (5)
BITS Jobs
Build Image on Host
Deobfuscate/Decode Files or Information
Deploy Container
Direct Volume Access
Domain Policy Modification (2)
Execution Guardrails (1)
Exploitation for Defense Evasion
File and Directory Permissions Modification (2)
Hide Artifacts (9)
Hijack Execution Flow (11)
Impair Defenses (9)
Indicator Removal on Host (6)
Indirect Command Execution
Masquerading (7)
Modify Authentication Process (4)
Modify Cloud Compute Infrastructure (4)
Modify Registry
Modify System Image (2)
Network Boundary Bridging (1)
Obfuscated Files or Information (6)
Pre-OS Boot (5)
Process Injection (11)
Reflective Code Loading
Rogue Domain Controller
Rootkit
Signed Binary Proxy Execution (13)
Signed Script Proxy Execution (1)
Subvert Trust Controls (6)
Template Injection
Traffic Signaling (1)
Trusted Developer Utilities Proxy Execution (1)
Unused/Unsupported Cloud Regions
Use Alternate Authentication Material (4)
Valid Accounts (4)
Virtualization/Sandbox Evasion (3)
Weaken Encryption (2)
XSL Script Processing
Credential Access
15 techniques
Adversary-in-the-Middle (2)
Brute Force (4)
Credentials from Password Stores (5)
Exploitation for Credential Access
Forced Authentication
Forge Web Credentials (2)
Input Capture (4)
Modify Authentication Process (4)
Network Sniffing
OS Credential Dumping (8)
Steal Application Access Token
Steal or Forge Kerberos Tickets (4)
Steal Web Session Cookie
Two-Factor Authentication Interception
Unsecured Credentials (7)
Discovery
29 techniques
Account Discovery (4)
Application Window Discovery
Browser Bookmark Discovery
Cloud Infrastructure Discovery
Cloud Service Dashboard
Cloud Service Discovery
Cloud Storage Object Discovery
Container and Resource Discovery
Domain Trust Discovery
File and Directory Discovery
Group Policy Discovery
Network Service Scanning
Network Share Discovery
Network Sniffing
Password Policy Discovery
Peripheral Device Discovery
Permission Groups Discovery (3)
Process Discovery
Query Registry
Remote System Discovery
Software Discovery (1)
System Information Discovery
System Location Discovery (1)
System Network Configuration Discovery (1)
System Network Connections Discovery
System Owner/User Discovery
System Service Discovery
System Time Discovery
Virtualization/Sandbox Evasion (3)
Lateral Movement
9 techniques
Exploitation of Remote Services
Internal Spearphishing
Lateral Tool Transfer
Remote Service Session Hijacking (2)
Remote Services (6)
Replication Through Removable Media
Software Deployment Tools
Taint Shared Content
Use Alternate Authentication Material (4)
Collection
17 techniques
Adversary-in-the-Middle (2)
Archive Collected Data (3)
Audio Capture
Automated Collection
Browser Session Hijacking
Clipboard Data
Data from Cloud Storage Object
Data from Configuration Repository (2)
Data from Information Repositories (3)
Data from Local System
Data from Network Shared Drive
Data from Removable Media
Data Staged (2)
Email Collection (3)
Input Capture (4)
Screen Capture
Video Capture
Command and Control
16 techniques
Application Layer Protocol (4)
Communication Through Removable Media
Data Encoding (2)
Data Obfuscation (3)
Dynamic Resolution (3)
Encrypted Channel (2)
Fallback Channels
Ingress Tool Transfer
Multi-Stage Channels
Non-Application Layer Protocol
Non-Standard Port
Protocol Tunneling
Proxy (4)
Remote Access Software
Traffic Signaling (1)
Web Service (3)
Exfiltration
9 techniques
Automated Exfiltration (1)
Data Transfer Size Limits
Exfiltration Over Alternative Protocol (3)
Exfiltration Over C2 Channel
Exfiltration Over Other Network Medium (1)
Exfiltration Over Physical Medium (1)
Exfiltration Over Web Service (2)
Scheduled Transfer
Transfer Data to Cloud Account
Impact
13 techniques
Account Access Removal
Data Destruction
Data Encrypted for Impact
Data Manipulation (3)
Defacement (2)
Disk Wipe (2)
Endpoint Denial of Service (4)
Firmware Corruption
Inhibit System Recovery
Network Denial of Service (2)
Resource Hijacking
Service Stop
System Shutdown/Reboot

MITRE Engage

Prepare	Expose		Affect			Elicit		Understand
Plan	**Collect**	**Detect**	**Prevent**	**Direct**	**Disrupt**	**Reassure**	**Motivate**	**Analyze**
Cyber Threat Intelligence	API Monitoring	Introduced Vulnerabilities	Baseline	Attack Vector Migration	Isolation	Application Diversity	Application Diversity	After-Action Review
Engagement Environment	Network Monitoring	Lures	Hardware Manipulation	Email Manipulation	Lures	Artifact Diversity	Artifact Diversity	Cyber Threat Intelligence
Gating Criteria	Software Manipulation	Malware Detonation	Isolation	Introduced Vulnerabilities	Network Manipulation	Burn-In	Information Manipulation	Threat Model
Operational Objective	System Activity Monitoring	Network Analysis	Network Manipulation	Lures	Software Manipulation	Email Manipulation	Introduced Vulnerabilities	
Persona Creation			Security Controls	Malware Detonation		Information Manipulation	Malware Detonation	
Storyboarding				Network Manipulation		Network Diversity	Network Diversity	
Threat Model				Peripheral Management		Peripheral Management	Personas	
				Security Controls		Pocket Litter		
				Software Manipulation				

#5 Threat Landscape

ENISA Threat Landscape 2018



ENISA Threat Landscape 2020

ENISA Threat Landscape 2021

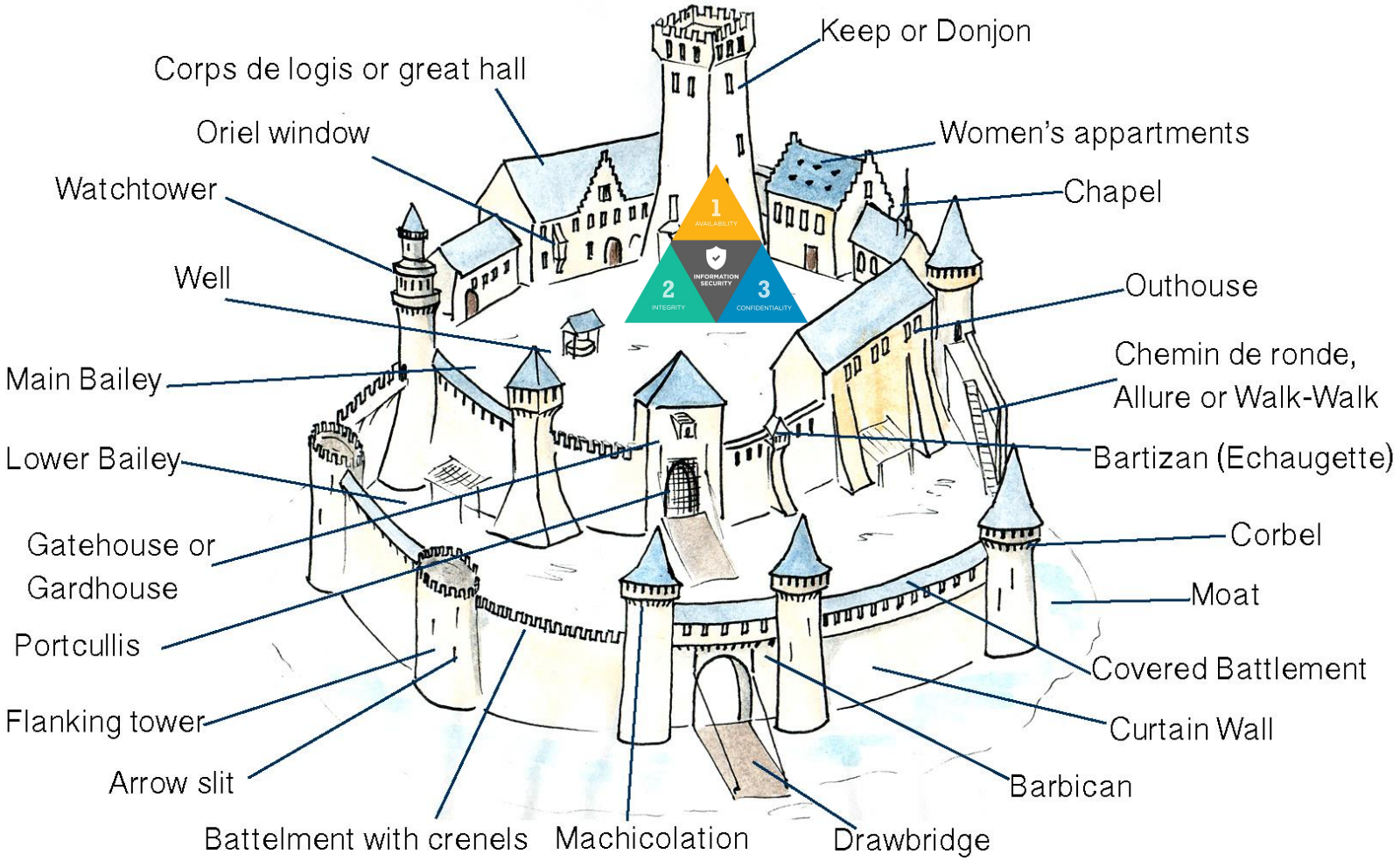
- Ransomware
- Malware
- Cryptojacking
- Email related threats
- Threats against data
- Threats against availability and integraity
- Disinformation – misinformation
- Non-malicious threats

Software Security

One important root cause of security issues

Information Security: CIA

People Think Security is …

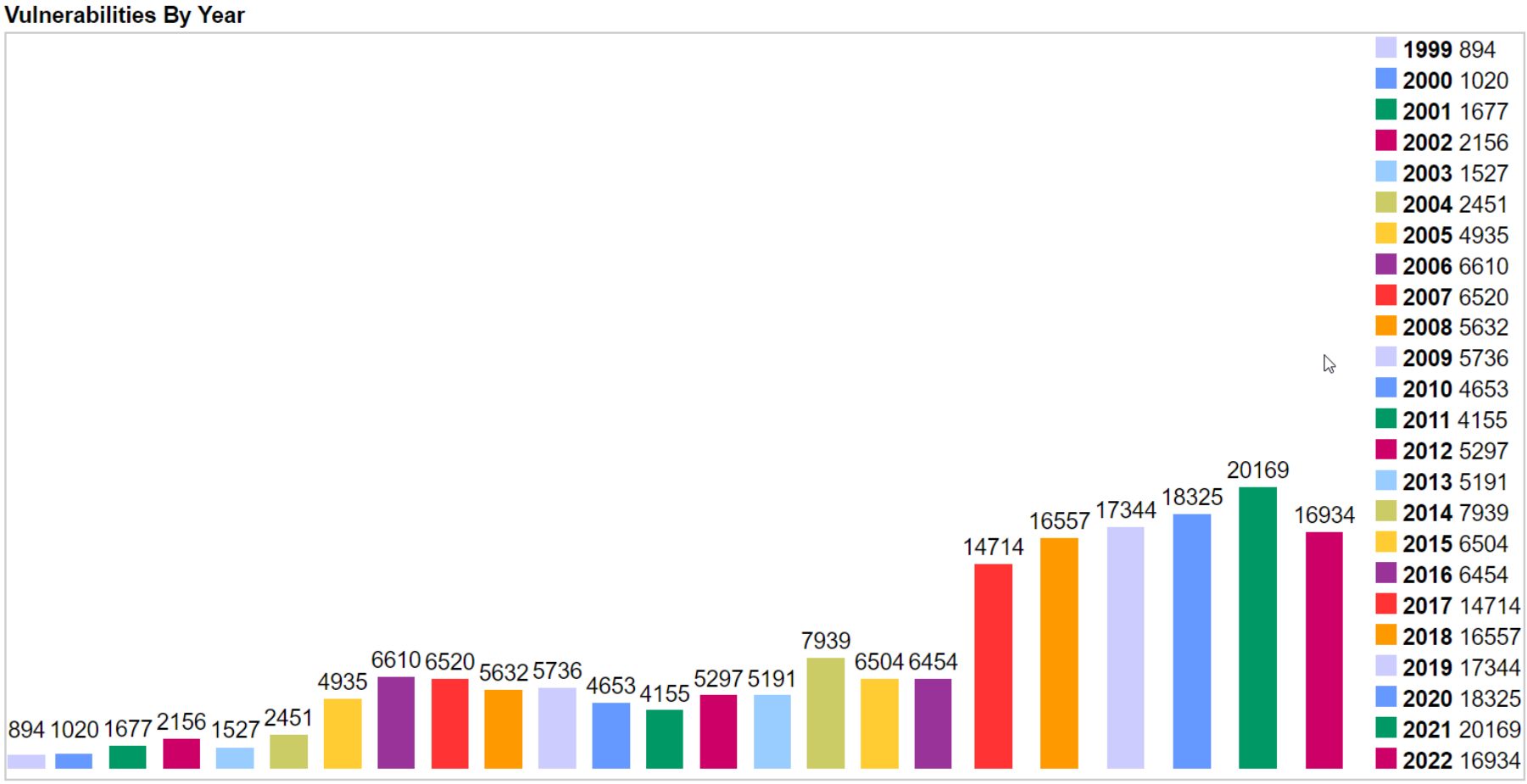


Source: https://www.roadtripsaroundtheworld.com/a-pocket-guide-to-medieval-castle-vocabulary/

A Typical Case on Security

Source: http://ithare.com/64-network-dos-and-donts-for-multi-player-game-developers-part-viib-security-concluded/

CVE: Common Vulnerabilities and Exposures



Source: https://www.cvedetails.com/browse-by-date.php

Improve Software Security

- Write clean codes
- Write bug-free codes
- Find (and then patch) bugs (automatically!)

Write Clean Codes

Writing Clean Codes

- Meaningful names
- Managable functions
- Good & correct comments
- Coding style and format
- Error handling
- Unit tests

Meaningful Names

- The variable interprets itself

```
int d;  // days since last modification
```

- Distinguish between names

```
int copyChars(char *a1, char *a2);
```

Managable Functions

- Small!
- Do one thing
- Liminted number of arguments (<3 ?)
- No side effects

Function: No Side Effects

```
1: boolean checkPassword(String userName, String password) {
2:   User user = UserGateway.findByName(userName);
3:   if (user != User.NULL) {
4:     String codedPhrase = user.getPhraseEncodedByPassword();
5:     String phrase = crypto.decrypt(codedPhrase, password);
6:     if ("Valid Password".equals(phrase)) {
7:         Session.initialize();
8:         return true;
9:     }
A:   }
B:   return false;
C: }
```

Comments

- Good **code** is just like good **joke**
  -- it needs no explanation

```
int main() {
    int a,b;          // declare two integers a, b
    cin>>a>>b;        // read user inputs for a and b
    cout<<a*a+b*b; // display a^2 + b^2
    return 0;         // exit with success
}
```

Error Handling

- Always perform the check
- Exceptions and return codes
- Do not return NULL
- Do not pass NULL

Write Bug-Free Codes

Writing Bug-Free Codes

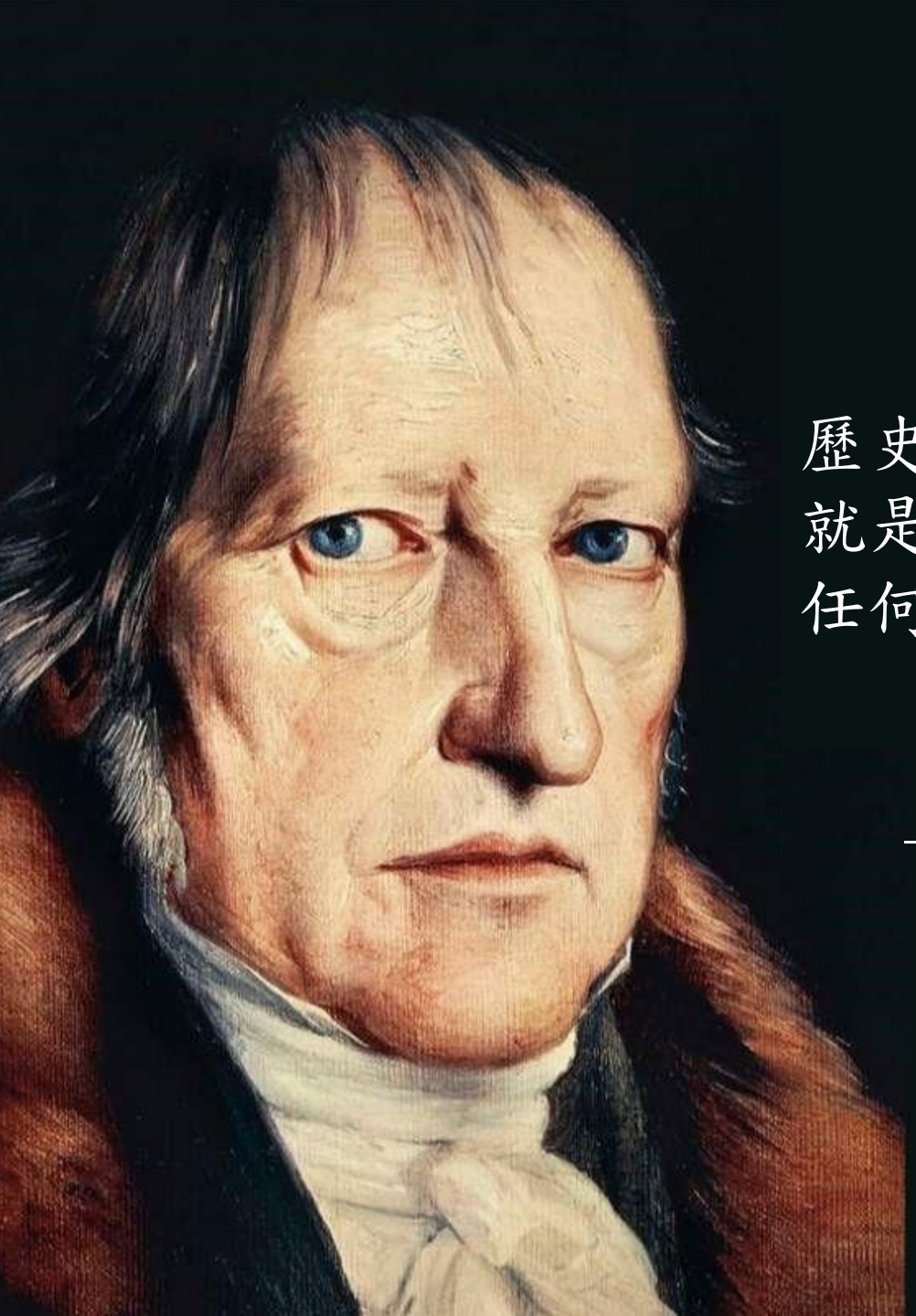
- Common mistakes
- Enable all compiler warnings
- ASSERT!
- Fortify source codes
- Many other guidelines

Common Mistakes: OWASP List of Vulnerabilities (60+)

Allowing Domains or Accounts to Expire
Buffer Overflow
Business logic vulnerability
CRLF Injection
CSV Injection
Catch NullPointerException
Covert storage channel
Deserialization of untrusted data
Directory Restriction Error
Doubly freeing memory
Empty String Password
Expression Language Injection
Full Trust CLR Verification issue Exploiting Passing Reference Types by Reference
Heartbleed Bug
Improper Data Validation
Improper pointer subtraction
Information exposure through query strings in url by Robert Gilbert

Common Mistakes: CWE Top 25 (2019)

Rank	ID	Name	Score
[1]	CWE-119	Improper Restriction of Operations within the Bounds of a Memory Buffer	75.56
[2]	CWE-79	Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')	45.69
[3]	CWE-20	Improper Input Validation	43.61
[4]	CWE-200	Information Exposure	32.12
[5]	CWE-125	Out-of-bounds Read	26.53
[6]	CWE-89	Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')	24.54
[7]	CWE-416	Use After Free	17.94
[8]	CWE-190	Integer Overflow or Wraparound	17.35
[9]	CWE-352	Cross-Site Request Forgery (CSRF)	15.54
[10]	CWE-22	Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal')	14.10
[11]	CWE-78	Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection')	11.47
[12]	CWE-787	Out-of-bounds Write	11.08
[13]	CWE-287	Improper Authentication	10.78
[14]	CWE-476	NULL Pointer Dereference	9.74
[15]	CWE-732	Incorrect Permission Assignment for Critical Resource	6.33
[16]	CWE-434	Unrestricted Upload of File with Dangerous Type	5.50
[17]	CWE-611	Improper Restriction of XML External Entity Reference	5.48
[18]	CWE-94	Improper Control of Generation of Code ('Code Injection')	5.36
[19]	CWE-798	Use of Hard-coded Credentials	5.12
[20]	CWE-400	Uncontrolled Resource Consumption	5.04
[21]	CWE-772	Missing Release of Resource after Effective Lifetime	5.04
[22]	CWE-426	Untrusted Search Path	4.40
[23]	CWE-502	Deserialization of Untrusted Data	4.30
[24]	CWE-269	Improper Privilege Management	4.23
[25]	CWE-295	Improper Certificate Validation	4.06



歷史給我們唯一的教訓，
就是我們無法從歷史中得到
任何教訓。

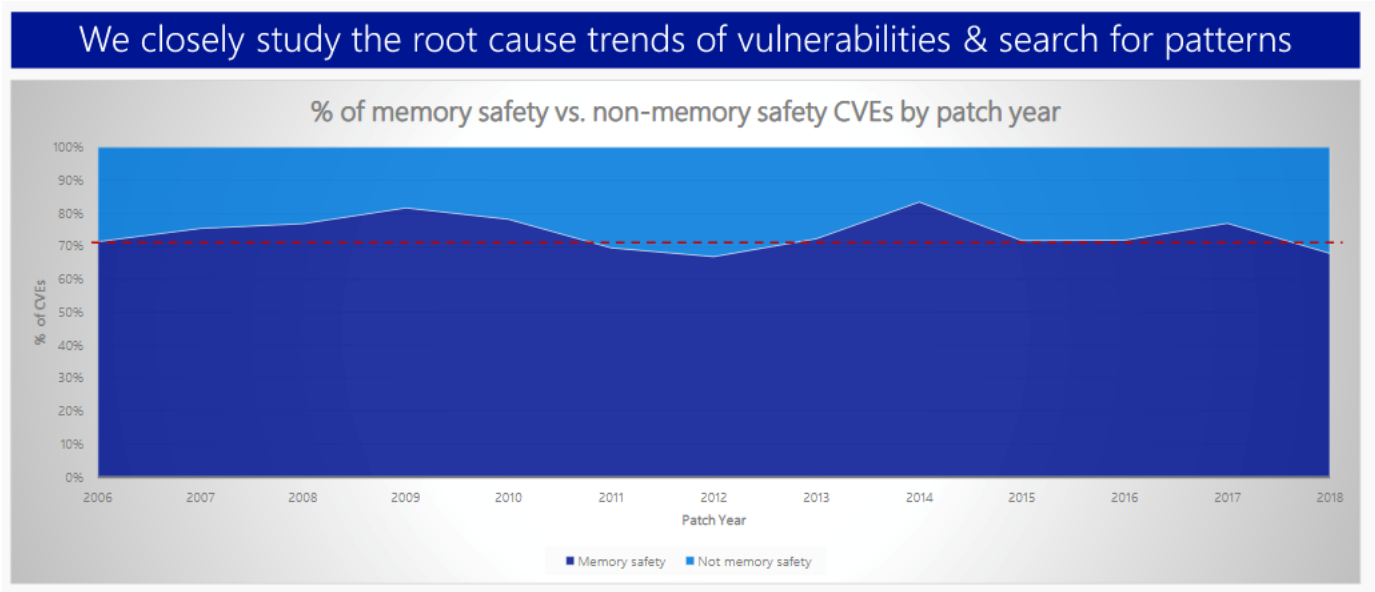
—日耳曼哲學家 黑格爾

Common Mistakes: CWE Top 25 (2022)

Rank	ID	Name	Score
[1]	CWE-787	Out-of-bounds Write	64.2
[2]	CWE-79	Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')	45.97
[3]	CWE-89	Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')	22.11
[4]	CWE-20	Improper Input Validation	20.63
[5]	CWE-125	Out-of-bounds Read	17.67
[6]	CWE-78	Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection')	17.53
[7]	CWE-416	Use After Free	15.5
[8]	CWE-22	Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal')	14.08
[9]	CWE-352	Cross-Site Request Forgery (CSRF)	11.53
[10]	CWE-434	Unrestricted Upload of File with Dangerous Type	9.56
[11]	CWE-476	NULL Pointer Dereference	7.15
[12]	CWE-502	Deserialization of Untrusted Data	6.68
[13]	CWE-190	Integer Overflow or Wraparound	6.53
[14]	CWE-287	Improper Authentication	6.35
[15]	CWE-798	Use of Hard-coded Credentials	5.66
[16]	CWE-862	Missing Authorization	5.53
[17]	CWE-77	Improper Neutralization of Special Elements used in a Command ('Command Injection')	5.42
[18]	CWE-306	Missing Authentication for Critical Function	5.15
[19]	CWE-119	Improper Restriction of Operations within the Bounds of a Memory Buffer	4.85
[20]	CWE-276	Incorrect Default Permissions	4.84
[21]	CWE-918	Server-Side Request Forgery (SSRF)	4.27
[22]	CWE-362	Concurrent Execution using Shared Resource with Improper Synchronization ('Race Condition')	3.57
[23]	CWE-400	Uncontrolled Resource Consumption	3.56
[24]	CWE-611	Improper Restriction of XML External Entity Reference	3.38
[25]	CWE-94	Improper Control of Generation of Code ('Code Injection')	3.32

Microsoft: Product CVE Statistics

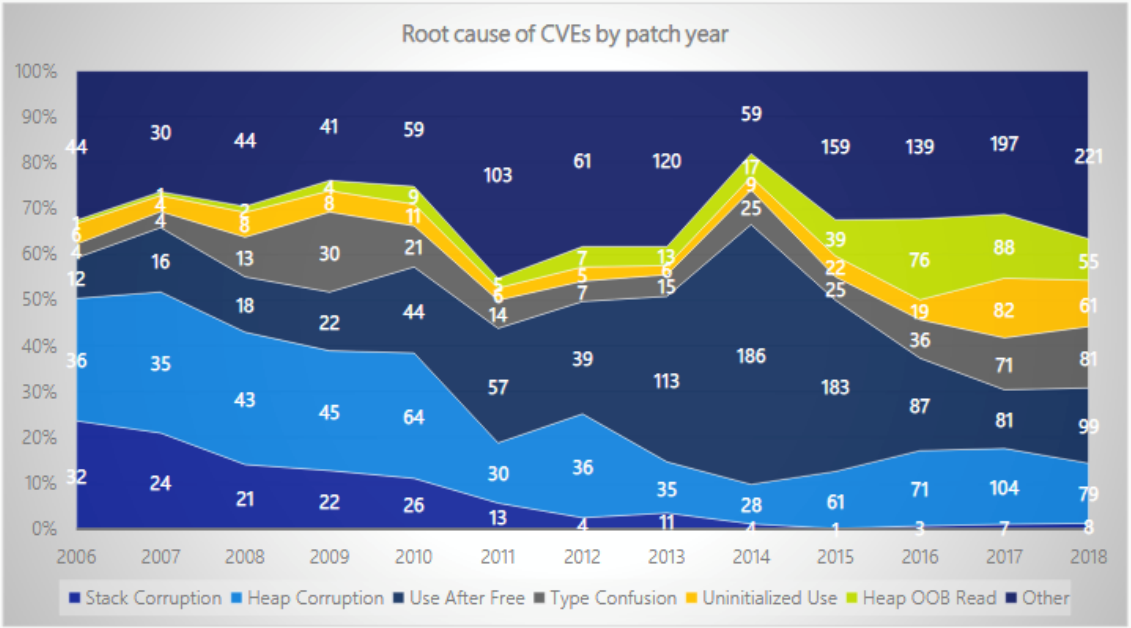
- 70%+ CVEs are memory safety bugs



Source: ZDNet https://www.zdnet.com/article/microsoft-70-percent-of-all-security-bugs-are-memory-safety-issues/

Microsoft: Category Breakdown

- Heap++, but Stack--



Stack corruptions are essentially dead

Use after free spiked in 2013-2015 due to web browser UAF, but was mitigated by Mem GC

Heap out-of-bounds read, type confusion, & uninitialized use have generally increased

Spatial safety remains the most common vulnerability category (heap out-of-bounds read/write)

Top root causes since 2016:

#1: heap out-of-bounds | #2: use after free | #3: type confusion | #4: uninitialized use

Source: ZDNet https://www.zdnet.com/article/microsoft-70-percent-of-all-security-bugs-are-memory-safety-issues/

Enable ALL Compiler Warnings

1

```
if(ch = 'a') {
        printf("Oh ... You typed 'a'\n");
}
```

```
huangant@rhino: ~ — 80×16
codes$ gcc cmp.c
cmp.c:6:8: warning: using the result of an assignment as a condition without
      parentheses [-Wparentheses]
        if(ch = 'a') {
           ~~~^~~~~
cmp.c:6:8: note: place parentheses around the assignment to silence this warning
        if(ch = 'a') {
              ^
           (       )
cmp.c:6:8: note: use '==' to turn this assignment into an equality comparison
        if(ch = 'a') {
              ^
              ==
1 warning generated.
codes$
```

Enable ALL Compiler Warnings

2

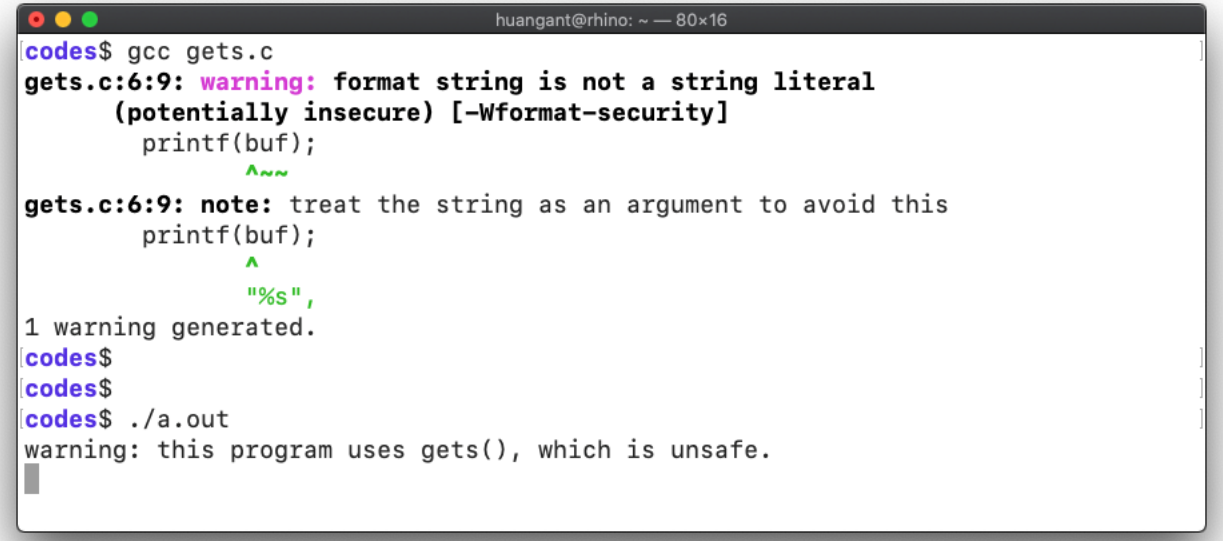
```
ch = getchar();
if('a' == ch ^ 0x20) {
    printf("Matched ... %x %x %x\n", 'a', ch, ch ^ 0x20);
}
```

```
huangant@rhino: ~ — 80×16
codes$ gcc and.c
and.c:7:15: warning: ^ has lower precedence than ==; == will be evaluated first
      [-Wparentheses]
        if('a' == ch ^ 0x20) {
           ~~~~~~~~~~^
and.c:7:15: note: place parentheses around the '==' expression to silence this
      warning
        if('a' == ch ^ 0x20) {
                     ^
           (         )
and.c:7:15: note: place parentheses around the ^ expression to evaluate it first
        if('a' == ch ^ 0x20) {
                     ^
                  (        )
1 warning generated.
codes$
```

Enable ALL Compiler Warnings

3

```
char buf[64];
gets(buf);
printf(buf);
```



ASSERT!

- Ensure that everything goes well

```
a = malloc(16);
assert(a != NULL);
```

```
huangant@zoolab: ~/www/snsl/chuang/courses/unixprog — 70×5
codes$ ./a.out
Assertion failed: (a != NULL), function main, file assert1.c, line 8.
Abort trap: 6
codes$
```

```
char getChar(const char *buf, size_t sz, int idx) {
  assert(buf != NULL && idx >= 0 && idx < sz);
  return buf[idx];
}
```

```
huangant@zoolab: ~/www/snsl/chuang/courses/unixprog — 70×5
codes$ ./a.out
Assertion failed: (buf != NULL && idx >= 0 && idx < sz), function getC
har, file assert2.c, line 6.
Abort trap: 6
codes$
```

- For debug ONLY!
  - assert() could be removed in ***non-debug*** mode (e.g., -DNDEBUG)

ASSERT: USE WITH CARE!

- EOS Node Remote Code Execution Vulnerability

2 libraries/chain/webassembly/binaryen.cpp View

```
@@ -73,7 +73,7 @@ std::unique_ptr<wasm_instantiated_module_interface> binaryen_runtime::instantiat
73  73        table.resize(module->table.initial);
74  74        for (auto& segment : module->table.segments) {
75  75           Address offset = ConstantExpressionRunner<TrivialGlobalManager>(globals).visit(segment.offset).value.geti32();
76        -        assert(offset + segment.data.size() <= module->table.initial);
      76  +        FC_ASSERT(offset + segment.data.size() <= module->table.initial);
77  77           for (size_t i = 0; i != segment.data.size(); ++i) {
78  78              table[offset + i] = segment.data[i];
79  79           }
```

1 comment on commit `ea89dce`

guhe120 commented on `ea89dce` 5 hours ago

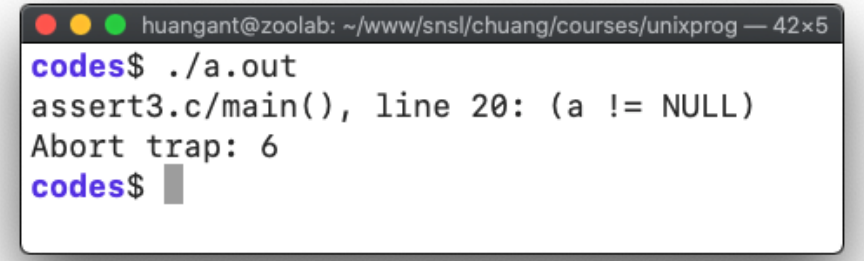
Hi, there is still some problem with this patch. in 32-bits process, offset + segment.data.size() could overflow and bypass the FC_ASSERT check

Source:
https://blog.360totalsecurity.com/en/eos-node-remote-code-execution-vulnerability-eos-wasm-contract-function-table-array-out-of-bounds/

ASSERT: Re-Implemented

```
#define MY_ASSERT(x)     \
        if (!(x)) {      \
                fprintf(stderr, "%s/%s(), line %d: (" #x ")\n",  \
                        __FILE__, __FUNCTION__, __LINE__);    \
                abort();  \
        } else
```

```
a = malloc(16);
MY_ASSERT(a != NULL);
```



Fortify Source Codes

- Many commercial tools
- The **lint** software – originates from a UNIX utility to examine C source codes
- Also available in different languages
  - Static code analysis tools

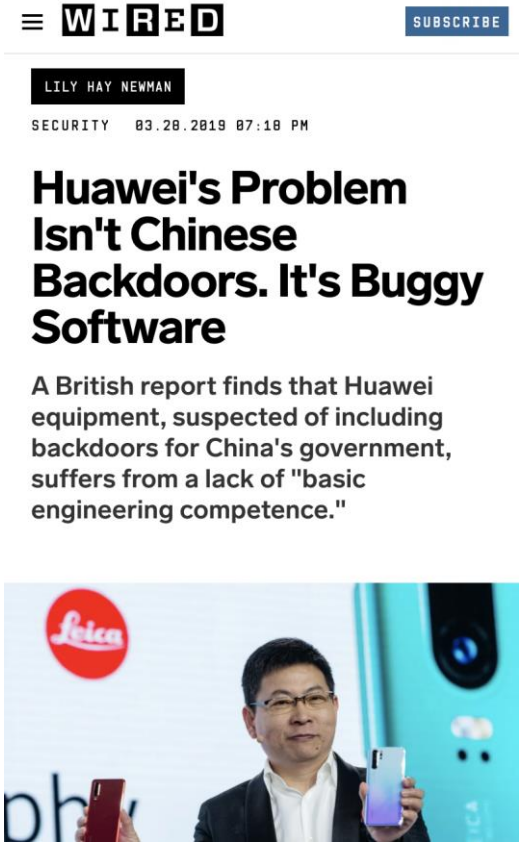
Other Guidelines

- CERT C Coding Standard (2016 Edition)
  - https://wiki.sei.cmu.edu/confluence/display/c
- MISRA C 2012
  - The Motor Industry Software Reliability Association @ UK
  - 汽車工業軟件可靠性協會
- Many books and online references
  - Writing { Clean | Solid | Secure | Bug-Free } Codes

Find Bugs

Backdoor vs. Bugdoor

Source: http://funnyenglish.altervista.org/17-idioms-with-animals-horse/



≡ WIRED SUBSCRIBE

LILY HAY NEWMAN

SECURITY 03.28.2019 07:18 PM

Huawei's Problem Isn't Chinese Backdoors. It's Buggy Software

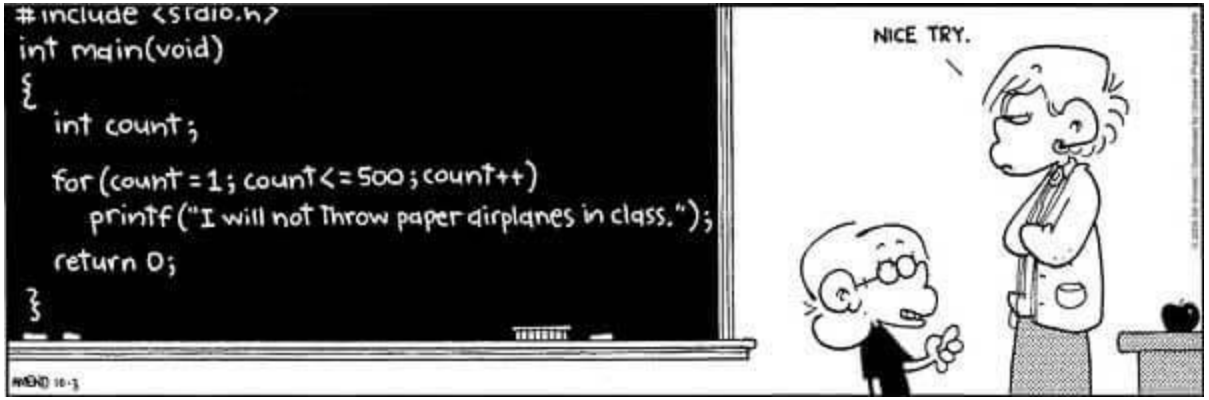
A British report finds that Huawei equipment, suspected of including backdoors for China's government, suffers from a lack of "basic engineering competence."

Source: https://www.wired.com/story/huawei-threat-isnt-backdoors-its-bugs/

Finding Bugs



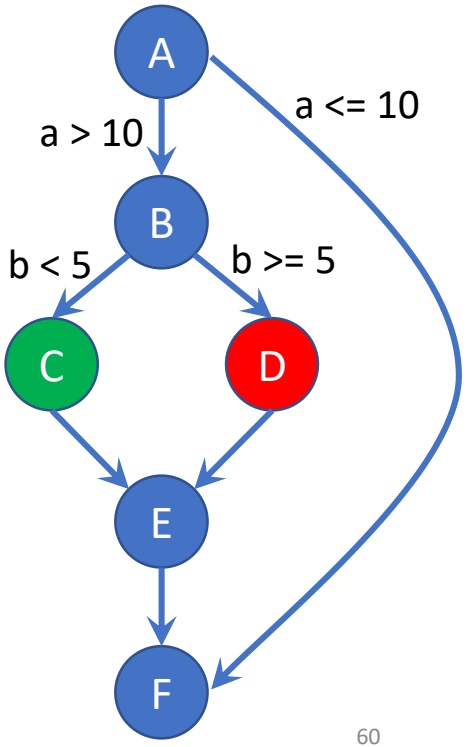

- LOC?
  - OS kernel: Linux and Windows
  - Web software: Apache, Chrome, and Firefox
- Usually we want to do it *automatically* (or *programmatically*)!



How to Automate?

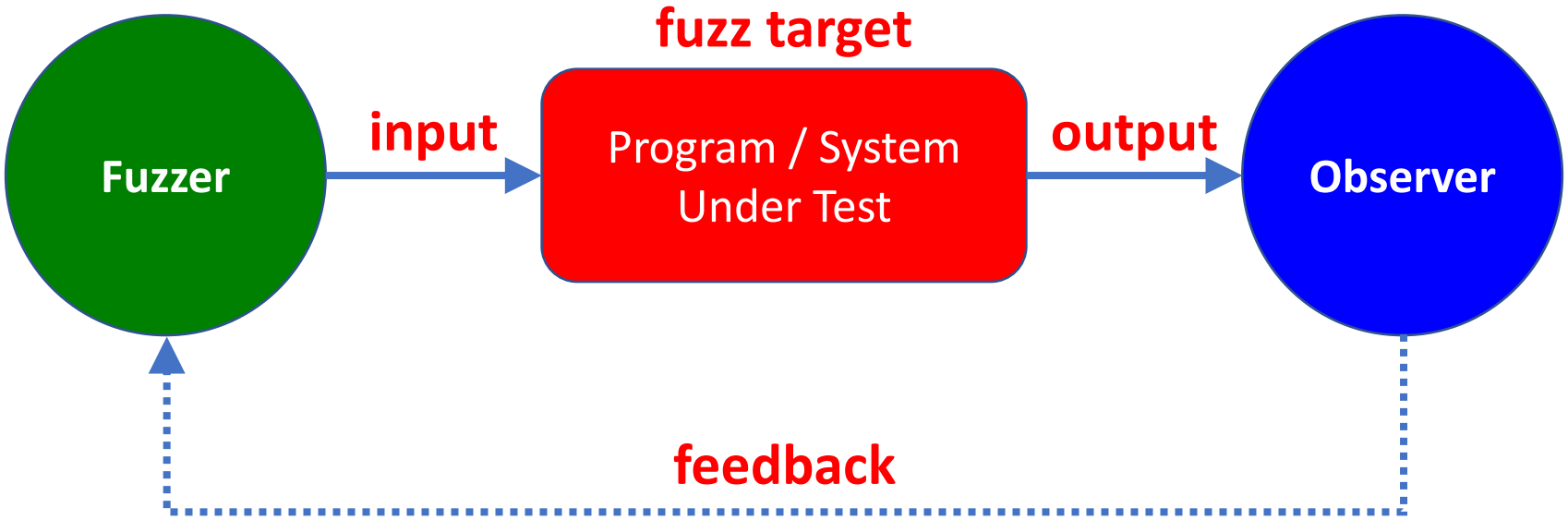
- Visit everywhere in the codes
- Symbolic execution vs. Fuzzing
- Symbolic execution
  - Solving constraints
  - Constraints for C: a > 10 && b < 5
  - Constraints for D: a > 10 && b >= 5
  - Constraints for E: a > 10 && b = ?
- Fuzzing
  - Randomly try a and b

```
1: if (a > 10) {
2:   if (b < 5) {
3:     ok();
4:   } else {
5:     bug();
6:   }
7:
8: }
```



Fuzz Testing (Fuzzing)

- An automated process to find bugs / vulnerabilities



Fuzz Testing (Fuzzing) (Cont'd)

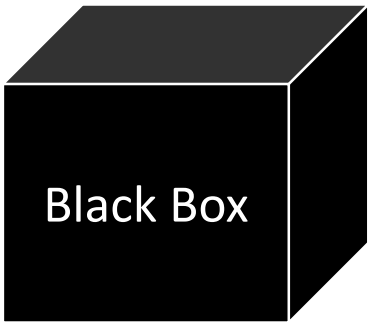
- Fuzzing is different from unit testing

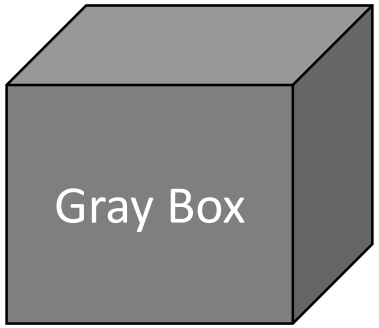
- Unit testing
  - **Known input** + **Expected output**
  - Good for regression testing

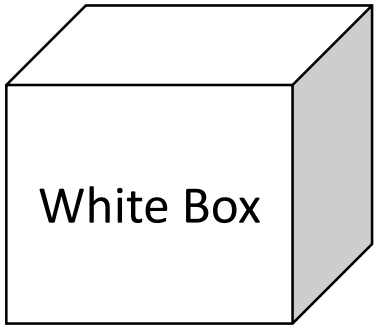
- Fuzz testing (fuzzing):
  - **Random input** + **Unexpected crash**
  - Good for security testing

Type of Fuzzers: Knowledge about the Target

- Black box: Zero knowledge
- Gray box: Some knowledge
- White box: Full knowledge



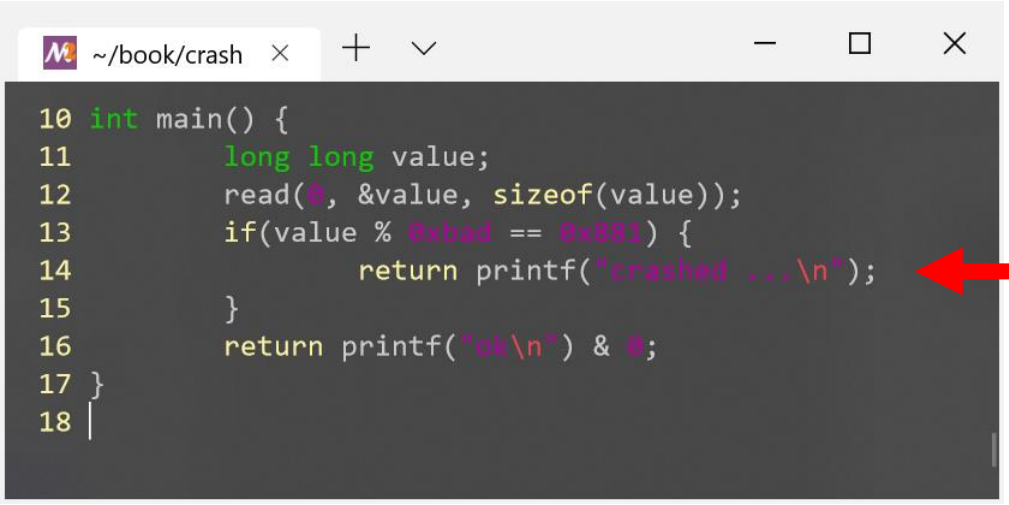




Type of Fuzzers: How Inputs are Generated

- Random Fuzzing: Knows nothing
- Template Fuzzing: Knows format, but no state
- Generation-based Fuzzing: Grammar + state
- Evolutionary Fuzzing: Based on feedback

How Fuzzing Works? – A Blackbox Fuzzer



Crash here!

Refer to: Lava-M dataset
https://sites.google.com/site/steelix2017/home/lava

How Fuzzing Works?

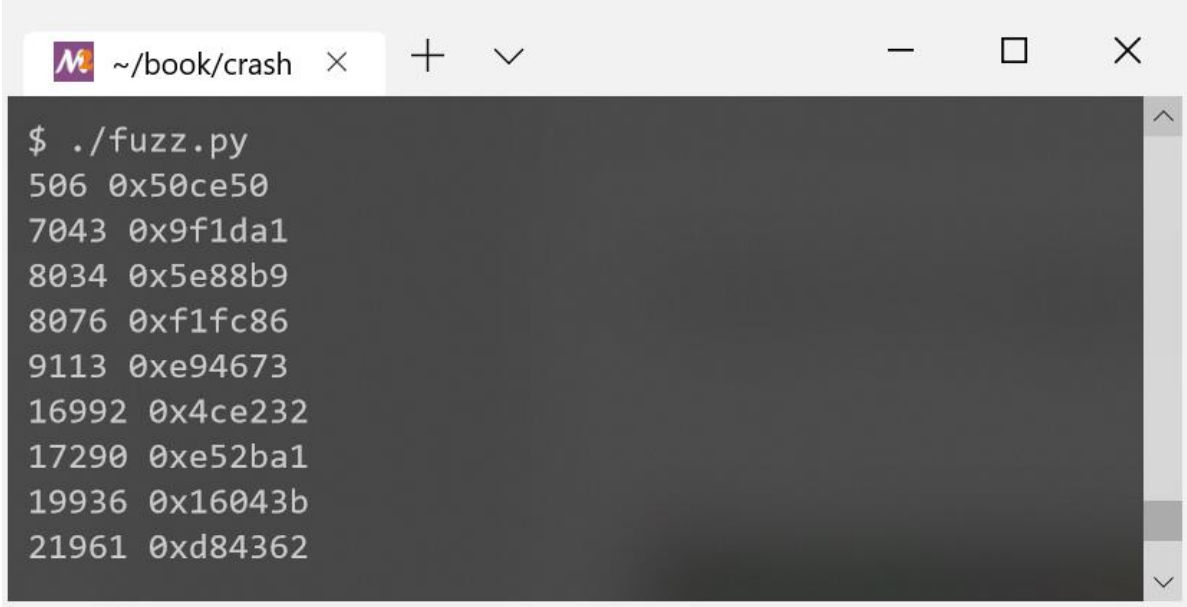
~/book/crash

```
10 int main() {
11         long long value;
12         read(0, &value, sizeof(value));
13         if(value % 0xbad == 0x881) {
14                 return printf("crashed ...\n");
15         }
16         return printf("ok\n") & 0;
17 }
18
```

~/book/crash

```
 4 import random
 5 import subprocess
 6
 7 for _ in range(1000000):
 8     m = random.randrange(2**24)
 9     n = bytes([ m & 0xff, (m>>8) & 0xff, (m>>16) & 0xff, (m>>24) & 0xff, 0, 0, 0, 0 ]);
10     r = subprocess.run(['./crash'], input=n, stdout=subprocess.PIPE, stderr=subprocess.PIPE);
11     if r.returncode != 0: print(_, hex(m));
12
```

How Fuzzing Works?



How Fuzzing Works? A Typical Evolutionary Graybox Fuzzer

- Input (seeds)
- Output (crashed seeds)
- Key functions
  - **choose_next**
  - **assign_energy**
  - **mutate_input**
  - **is_interesting**

Input: Seed inputs S

1: $Q_x = \phi$
2: $Q = S$
3: **if** $Q = \phi$ **then**
4: add an empty file to Q
5: **end if**
6: **repeat**
7: $s = choose\_next(Q)$
8: $p = assign\_energy(s)$
9: **for** $i = 1$ **to** p **do**
10: $s' = mutate\_input(s)$
11: **if** $fuzz(s')$ crashes **then**
12: add s' to Q_x
13: **else if** $is\_interesting(s')$ **then**
14: add s' to Q
15: **end if**
16: **end for**
17: **until** timeout or abort-signal received

Output: Crashed inputs Q_x

Bugs Found by Fuzzers

- AFL – Author's website
  - https://lcamtuf.coredump.cx/afl/#bugs
  - 400+ bugs
- AFL-CVE – Collected CVE from 3rd Parties
  - https://github.com/mrash/afl-cve
  - (2013-2017) 300+ bugs
- Syzbot – Linux kernel bugs
  - https://syzkaller.appspot.com/upstream
  - (2017—) 5000+ bugs

SSDLC Integration

SDLC

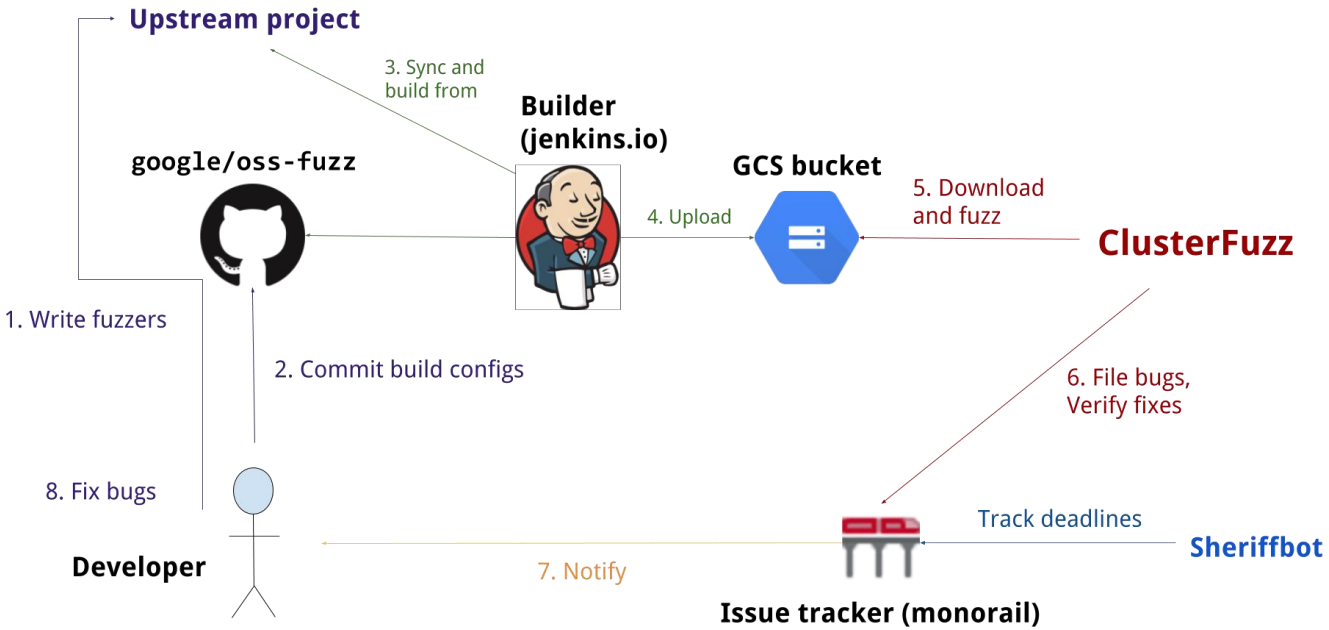
- Requirements
- Design
- Implementation/Coding
- Testing/Verification
- Deployment/Operation
- Maintenance

Secure SDLC

- *Add* security-related activities to all phases
- *Continuous* security integration
- Guidelines
  - MS SDL
  - NIST 800-64
  - OWASP CLASP

SSDLC: Google's OSS Fuzz

- Continuous fuzzing for open source software
- https://github.com/google/oss-fuzz



Learning Security

Resources

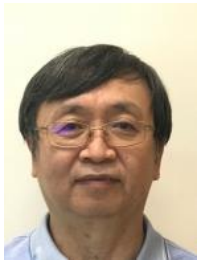
- **C**ourses
- **C**lubs / **C**ommunities / **C**ompanies
- **C**ompetitions
- **C**ertificates

Security Professors in CS Dept.

謝續平
網路安全、企業資安

曾文貴
密碼學、資訊安全、網路安全

林盈達
網路安全



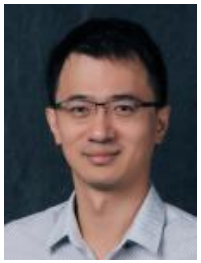
蔡錫鈞
密碼學

王協源
網路安全

楊武
系統安全

黃世昆
網路安全

吳育松
系統安全



吳凱強
硬體安全

李奇育
網路和系統安全

黃俊穎
網路安全

Security Courses in CS Dept.

110下	5252	IOC5167		軟體測試	30	70	R56-CS105[GF]	3.00	3.00	黃世昆	選修
		本院學生優先；[另外1小時(第3節)以非同步上課方式the third period of class will be held asynchronously]；資電亥客與安全碩士學位學程必選修課程									
110下	5277	IOE5118		行動網路安全 (英文授課) 英文授課	40	26	R34n-ED202[GF]	3.00	3.00	李奇育	選修
		本院學生優先；英文授課[English Medium Course]；資電亥客與安全碩士學位學程必選修課程；									
111上	535605	CSIC30093		網路安全 英文授課	88	-	T34n-EC115[GF]	3.00	3.00	謝續平	選修
		本院學生優先；博士班資格考考核科目；資安學程必選修課程；[English Medium Course]									
110上	1154	DCP1323		密碼學概論	115	151	T2F56-ED117[GF]	3.00	3.00	曾文貴	選修
		外系生請於開學上網加選；學士班七大主題學程；資工系與電資學士班學生優先選課									
111上	535512	CSIC30085		程式安全	45	-	F234-EC324[GF]	3.00	3.00	黃俊穎	選修
		本院學生優先；資安學程必選修課程									
110上	5254	IOC5063		密碼理論	45	22	W34F2-ED305[GF]	3.00	3.00	曾文貴	選修
		本院學生優先									
110下	1159	DCP3123		電腦安全總整與實作(英文授課) 英文授課	60	55	M34W2-EC114[GF]	3.00	3.00	李奇育	選修
		外系生請於開學上網加選；須修過基礎程式設計才可修電腦安全總整與實作(英文授課)；資工系與電資學士班學生優先選課；學士班七大主題學程；原課名「電腦安全概論」與「電腦安全總整與實作」與新舊課程僅能採計一門為畢業學分，不能重覆採計；英文授課[English Medium Course]									
111上	515618	CSCS20008		企業網路安全	40	-	R34-EC329[GF]	3.00	3.00	謝續平	選修
		外系生請於開學上網加選；資工系與電資學士班學生優先選課；主開3年級；不開放大1及大2選課；[另外1小時(第3節)以非同步上課方式; the third period of class will be held asynchronously]									
110下	5250	IOC5129		容錯計算	60	15	W34-ED302[GF]	3.00	3.00	吳育松	選修
		本院學生優先[另外1小時(第3節)以非同步上課方式the third period of class will be held asynchronously]									

Courses

- Coursera
  - https://www.coursera.org/search?query=security
- PwnCollege
  - https://pwn.college/

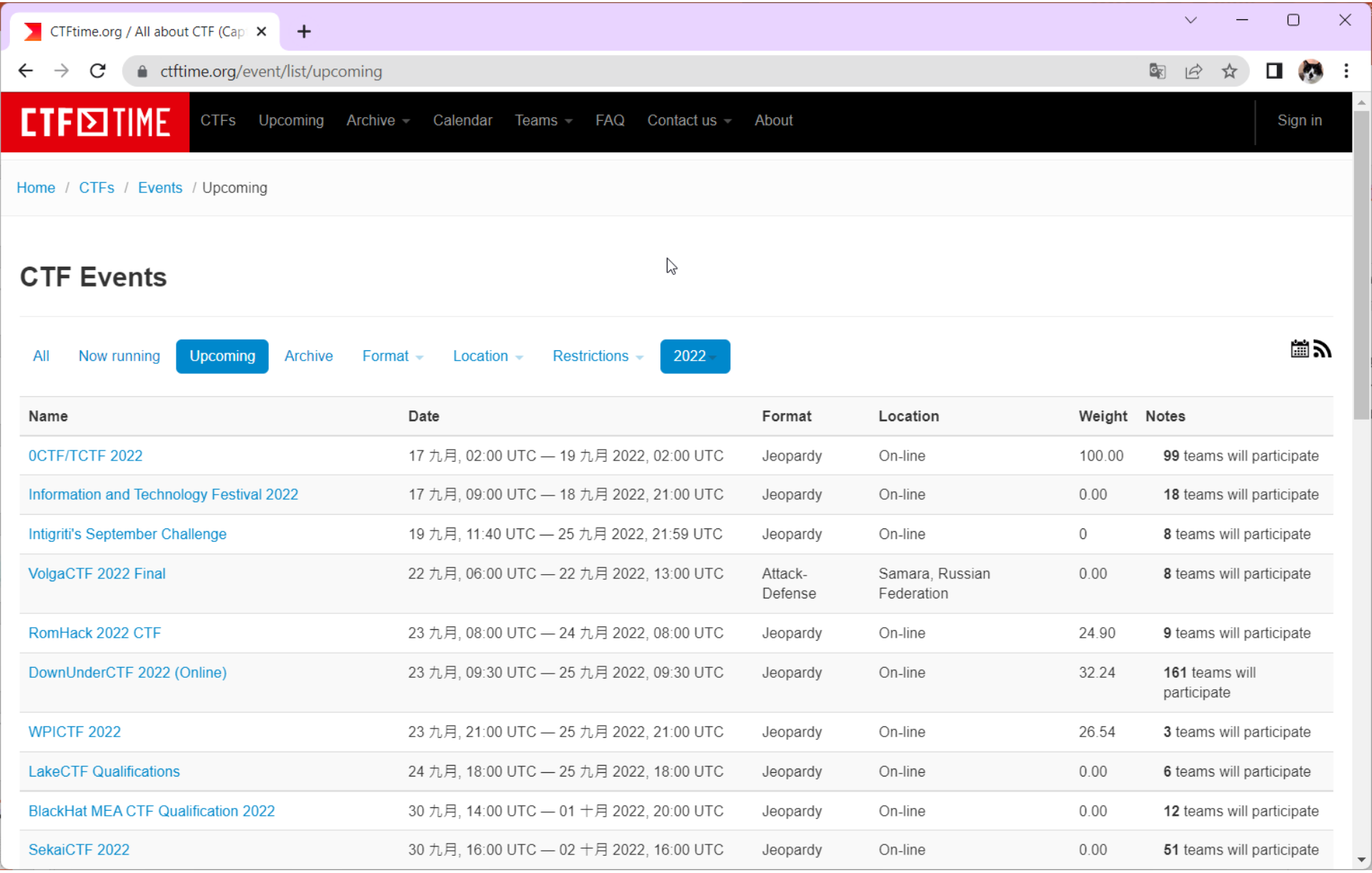
Clubs / Communities / Companies

- 各大學資安社
- AIS3 / HITCON / TDOHacker / UCCU
- 多到數不清的資安公司
  - 實習機會

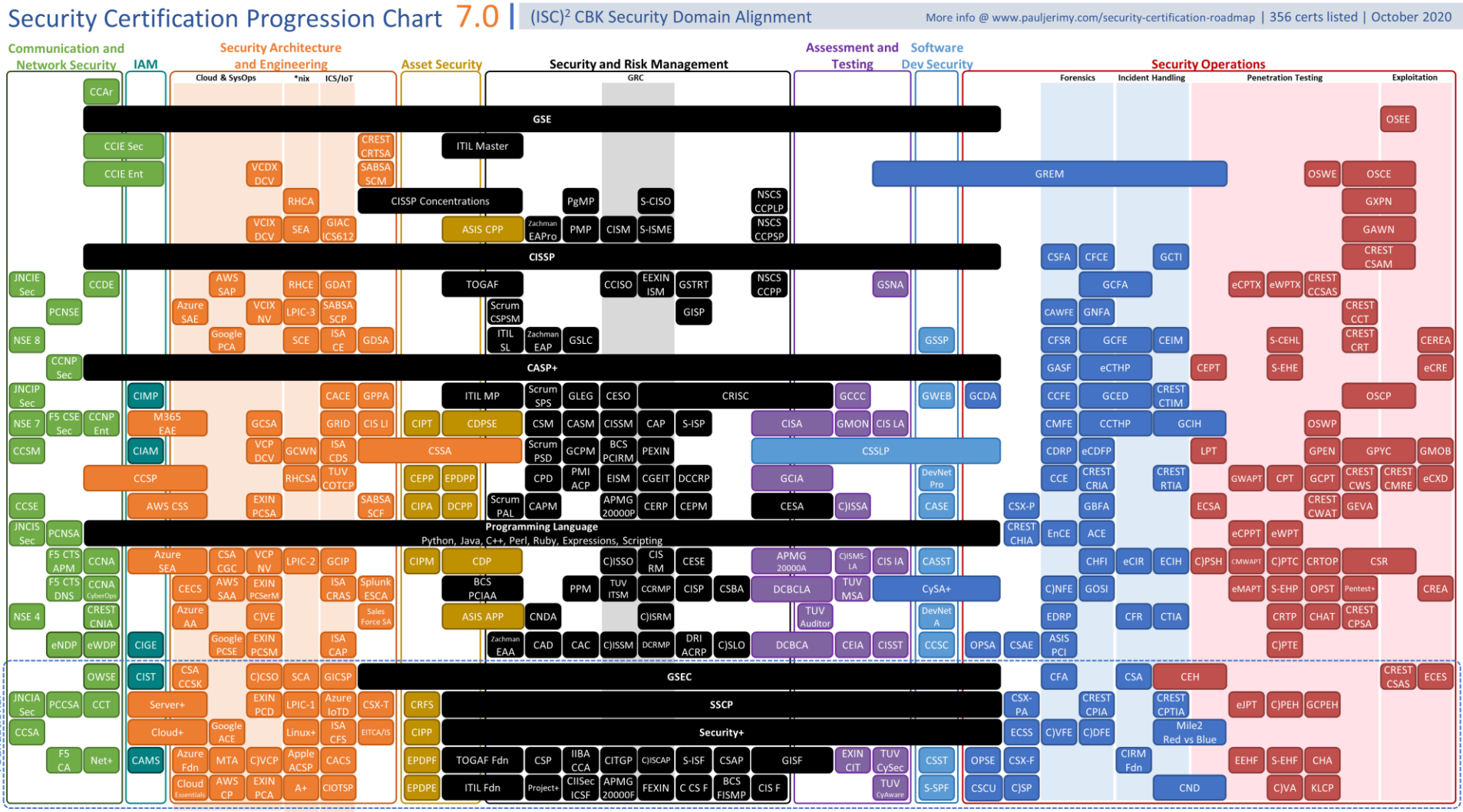
Competitions

- Domestic ONLY
  - 科技部/女婕思(三月報名)
  - AIS3/MyFirstCTF (四月報名)
  - 技服中心/金盾獎 (九月報名)
  - AIS3/EOF (十二月報名)

- International … Many online CTFs
  - PicoCTF - https://picoctf.org/
  - https://ctftime.org/event/list/upcoming


CTFtime.org / All about CTF (Cap
ctftime.org/event/list/upcoming
CTF TIME
CTFs
Upcoming
Archive
Calendar
Teams
FAQ
Contact us
About
Sign in
Home / CTFs / Events / Upcoming
CTF Events
All
Now running
Upcoming
Archive
Format
Location
Restrictions
2022
Name
Date
Format
Location
Weight
Notes
0CTF/TCTF 2022
17 九月, 02:00 UTC — 19 九月 2022, 02:00 UTC
Jeopardy
On-line
100.00
99 teams will participate
Information and Technology Festival 2022
17 九月, 09:00 UTC — 18 九月 2022, 21:00 UTC
Jeopardy
On-line
0.00
18 teams will participate
Intigriti's September Challenge
19 九月, 11:40 UTC — 25 九月 2022, 21:59 UTC
Jeopardy
On-line
0
8 teams will participate
VolgaCTF 2022 Final
22 九月, 06:00 UTC — 22 九月 2022, 13:00 UTC
Attack-Defense
Samara, Russian Federation
0.00
8 teams will participate
RomHack 2022 CTF
23 九月, 08:00 UTC — 24 九月 2022, 08:00 UTC
Jeopardy
On-line
24.90
9 teams will participate
DownUnderCTF 2022 (Online)
23 九月, 09:30 UTC — 25 九月 2022, 09:30 UTC
Jeopardy
On-line
32.24
161 teams will participate
WPICTF 2022
23 九月, 21:00 UTC — 25 九月 2022, 21:00 UTC
Jeopardy
On-line
26.54
3 teams will participate
LakeCTF Qualifications
24 九月, 18:00 UTC — 25 九月 2022, 18:00 UTC
Jeopardy
On-line
0.00
6 teams will participate
BlackHat MEA CTF Qualification 2022
30 九月, 14:00 UTC — 01 十月 2022, 20:00 UTC
Jeopardy
On-line
0.00
12 teams will participate
SekaiCTF 2022
30 九月, 16:00 UTC — 02 十月 2022, 16:00 UTC
Jeopardy
On-line
0.00
51 teams will participate

Certificates



Source: https://pauljerimy.com/security-certification-roadmap/

Other Resources

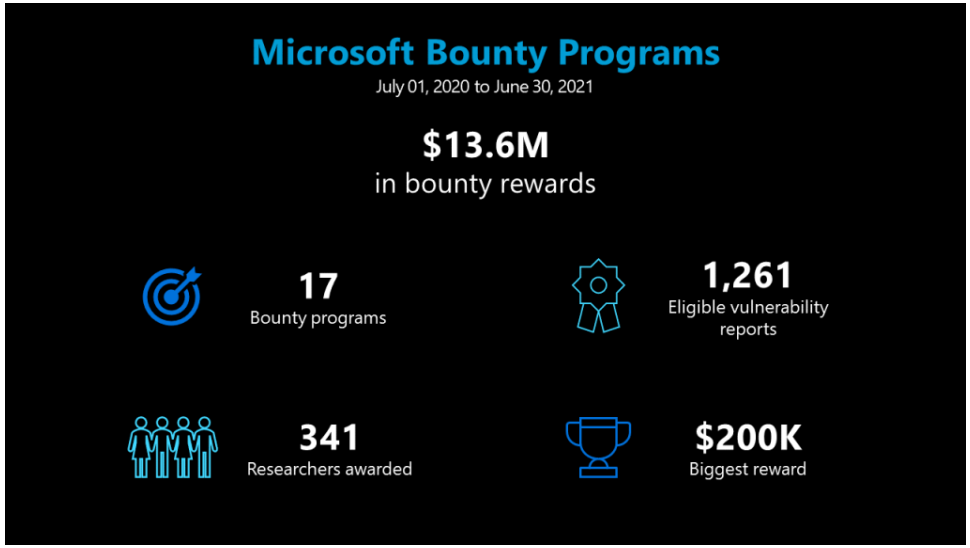
- DVWA
  - Damn Vulnerable Web Application
- OWASP Goat 系列
  - WebGoat, NodeGoat, PyGoat, AndroGoat
  - https://owasp.org/projects/
- Vulhub 弱點系列
  - https://github.com/vulhub/vulhub
  - https://www.vulnhub.com/
  - https://vulhub.org/ (對岸)

Bug Bounties

List of Bug Bounty/Crowdsourced Security Platforms

- ✔ HackerOne - www.hackerone.com
- ✔ Bugcrowd - www.bugcrowd.com
- ✔ Synack - www.synack.com/red-team
- ✔ Detectify - cs.detectify.com
- ✔ Cobalt - cobalt.io
- ✔ Open Bug Bounty - www.openbugbounty.org
- ✔ Zerocopter - www.zerocopter.com
- ✔ YesWeHack - www.yeswehack.com
- ✔ HackenProof - hackenproof.com
- ✔ Vulnerability Lab - www.vulnerability-lab.com
- ✔ FireBounty - firebounty.com
- ✔ BugBounty[.]jp - bugbounty.jp
- ✔ AntiHACK - www.antihack.me
- ✔ Intigriti - www.intigriti.com
- ✔ SafeHats - safehats.com
- ✔ RedStorm - www.redstorm.io
- ✔ Cyber Army ID - www.cyberarmy.id
- ✔ Yogosha - yogosha.com

Source: https://itblogr.com/list-of-bugbounty-croudsource-platforms/



Source: https://msrc-blog.microsoft.com/2021/07/08/microsoft-bug-bounty-programs-year-in-review-13-6m-in-rewards/

Source: https://hackerone.com/directory/programs?order_direction=DESC&order_field=average_bounty

Bug Bounties – HackerOne

Asset type

- (•) Any
- ( ) CIDR
- ( ) Domain
- ( ) iOS: App Store
- ( ) iOS: Testflight
- ( ) iOS: Ipa
- ( ) Android: Play Store
- ( ) Android: .apk
- ( ) Windows: Microsoft Store
- ( ) Source code
- ( ) Executable
- ( ) Hardware/IoT
- ( ) Other

PayPal	09 / 2018	1464	-	$1k-$3k
Nintendo	12 / 2016	92	$100	$1k-$2k
GitHub Security Lab	11 / 2019	0	$500	$1k
Zilliqa	09 / 2021	4	$100	$1k-$6k
Ruby on Rails	12 / 2015	63	$500	$1k
GitLab	02 / 2016	1104	-	$1k
IOVLabs	04 / 2018	17	$400	$1k
Xilinx, now part of AMD – Bug Bounty Program	11 / 2020	11	$50	$1k
MetaMask	06 / 2022	5	$50	$1k
Valve	05 / 2018	856	$100	$750
Zerly	12 / 2019	43	$50	$750
Payoneer	01 / 2022	17	$50	$750
curl	04 / 2019	37	$500	$700-$800

Source: https://www.microsoft.com/en-us/msrc/bounty?rtc=1

Bug Bounties – Microsoft

Cloud Programs

Program Name	Start date	Last Updated	End date	Eligible entries	Bounty Range
Microsoft Azure	2014-09-23	2021-10-18	Ongoing	Vulnerability reports on Microsoft Azure cloud services	Up to $60,000 USD
Microsoft Identity	2018-07-17	2019-10-23	Ongoing	Vulnerability reports on Identity services, including Microsoft Account, Azure Active Directory, or select OpenID standards.	Up to $100,000 USD
Xbox	2020-01-30	2020-01-30	Ongoing	Vulnerability reports on the Xbox Live network and services	Up to $20,000 USD
M365	2014-09-23	2019-08-05	Ongoing	Vulnerability reports on applicable Microsoft cloud services, including Office 365	Up to $20,000 USD
Microsoft Azure DevOps Services	2019-01-17	2019-01-17	Ongoing	Vulnerability reports on applicable Microsoft Azure DevOps Services	Up to $20,000 USD
Microsoft Dynamics 365 and Power Platform	2019-07-17	2022-04-14	Ongoing	Vulnerability reports on applicable Microsoft Dynamics 365 and Power Platform applications	Up to $20,000 USD
Microsoft .NET	2016-09-01	2020-11-20	Ongoing	Vulnerability reports on .NET Core and ASP.NET Core RTM and future builds (see link for program details)	Up to $15,000 USD

Platform Programs

Program Name	Start Date	Last Updated	End Date	Eligible Entries	Bounty Range
Microsoft Hyper-V	2017-05-31	2020-04-13	Ongoing	Critical remote code execution, information disclosure and denial of services vulnerabilities in Hyper-V	Up to $250,000 USD
Microsoft Windows Insider Preview	2017-07-26	2020-08-27	Ongoing	Critical and important vulnerabilities in Windows Insider Preview	Up to $100,000 USD
Microsoft Applications and On-Premises Servers	2021-03-24	2022-04-05	Ongoing	Critical and important vulnerabilities in Microsoft Applications and On-Premises Servers	Up to $30,000 USD

Summary

Takeaway Points

- Security is an endless arms race
- Software security is the fundamental!
- Mindset to improve software security
- Learning security for fun and profit!

Q & A

Thanks for your attention

chuang@cs.nctu.edu.tw